CONGESTION GAME-BASED TASK ALLOCATION FOR MULTI-ROBOT TEAMS

A Dissertation
Presented to
The Academic Faculty

By

Hai Zhong

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2020

CONGESTION GAME-BASED TASK ALLOCATION FOR MULTI-ROBOT TEAMS

Approved by:

Dr. Seth Hutchinson, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Samuel Coogan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Kyriakos Vamvoudakis
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: April 22, 2020

I think I'll stop here.

Andrew Wiles

To my parents

ACKNOWLEDGEMENTS

I would like to express my greatest gratitude towards my advisor, Professor Seth Hutchinson. I still remember the first time I met with Seth when I almost knew nothing about robotics. Seth lead me to the world of multi-robot research. Seth guide me through my two years' journey at Georgia Tech. Every time I meet with Seth, I grow up and learn new things. Seth's passion and enthusiasm for work, well-organization of the busy schedule, and clear insight of the big picture deeply inspire me. Seth is truly my role model.

I would also like to thank Professor Samuel Coogan and Professor Kyriakos Vamvoudakis, for serving in my thesis committee. I am also thankful to Professor Shuai Li for his comments on the thesis and bring me to great restaurants every time he visits Georgia Tech. I am grateful for Professor Guotong Zhou and Professor Fumin Zhang for their guidance and advice.

I want to thank my friends at Georgia Tech. First, thanks to Chao Tang, Ziyi Zhou and Jianning Cui. We cook countless delicious meals together and spend lots of happy times together. The happiness we share together throughout the past two years makes my study at Georgia Tech truly enjoyable. I would like to thank Zirui Xu, who give me advice when I have trouble. I would like to thank Sen Wang, Huizong Yang, Hongchang Zhu, Jinhao Jiang for all the happy times we spend together. I also want to thank Dr. Hongxiao Li and Dr. Li Sun. I am so grateful for taking me to the grocery stores every week and constantly taking care of my life. I owe a lot to you two. I also thank for Professor Feng Gao for taking care of me for my first year at Tech. I thank all the friends I made during my two year's study at Georgia Tech.

Finally, I would like to thank my parents. Their love is always my motivation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Multi-robot teams can complete complex missions that are not amenable to an individual robot. A team of heterogeneous robots with complementing capabilities is endowed with advantages to allow deep collaboration in dynamic and complicated environments. Multi-robot Task Allocation (MRTA) presents a fundamental for multi-robot system research. Despite the previous research efforts, there remains a knowledge gap in developing decentralized approaches for MRTA by viewing robots as resources and optimizing the distribution of robots to achieve the best overall performance at the system level. To address this knowledge gap, the objective of this research is to develop decentralized resource allocation algorithms to provide approximate solutions for the MRTA problem.

Both standard congestion game theory and weighted congestion game theory are exploited as the theoretical framework to formulate and solve the MRTA problems. Two types of resource allocation problems are considered, one has increasing marginal gain with respect to the number of participating robots, the other has decreasing marginal gain with respect to the number of participating robots. For MRTA problems with homogeneous robot teams, the sequential best response dynamics is integrated in the framework of standard congestion game theory. A concurrent version of best response dynamics with convergence guarantees is developed. In addition, a decentralized dual greedy algorithm is proposed and its convergence to a pure Nash equilibrium is proved. For MRTA problems with heterogeneous robot teams, the best sequential dynamics is shown to converge to pure Nash equilibrium in the framework of weighted congestion games. The suboptimality of the approximate solutions is discussed by $\lambda - \mu$ smoothness technique. Simulations and experiments using robots in the Robotarium are conducted to validate the effectiveness of the proposed algorithms.

CHAPTER 1
INTRODUCTION

1.1 Multi-Robot Task Allocation

Remarkable progress has been made in many areas of robotics-from microrobots for medicines to large robotic arms in building constructions, and from space robots for outer space explorations to underwater robots designed for deep-sea explorations [1]. Robotics are expected to transform people's daily life profoundly. For example, robots help people to combat the pandemic of 2019 novel coronavirus (COVID-19) by delivering food and contaminated waste, and taking care of patients (e.g., telemedicine and monitoring the health status) [2].

Multi-robot system (swarm robotics) research is a subarea of robotics research, which is recognized as one of the ten grand challenges for robotics [1]. Multi-robot teams will work with people to enrich the quality of life and transform future work, such as environment monitoring [3], and search and rescue [4]. Since tasks become more complicated and require different capabilities, a homogeneous robot team with a single type of robot may be insufficient to complete the task. Hence, it is imperative to develop robot teams composed of heterogeneous robots. For example, teams of robots with different sensing abilities could be distributed across a lake to monitor water quality [3]; groups of ground vehicles and quadrotors could be employed to perform collaborative mapping for an earthquake-damaged building [4].

MRTA is a fundamental problem in the field of multi-robot systems research. Since robot teams need to allocate tasks to team members before performing the task, it is inappropriate to rely solely on human operators to assign different tasks to robots due to the increasing number of robots in robot teams as well as the growing complexity of robot tasks. Previous work has focused on designing both centralized and decentralized MRTA

algorithms. When MRTA problems involve optimizing the sum of robots' utilities (when a robot's utility is fixed for a specific task) or the sum of robots' total traveling distance, there exist decentralized approaches providing approximate solutions [5]. However, there is a gap between centralized and decentralized approaches when the optimization problem being solved is a resource allocation problem. Resource allocation problems view robots as resources and strive to distribute robots to different tasks such that the overall performances are optimized. The objective of this thesis is to develop decentralized MRTA algorithms to provide approximate solutions to resource allocation problems for both homogeneous and heterogeneous robot teams. This is achieved by using congestion game theory.

1.2 Congestion Games

Congestion games were introduced by Rosenthal [6]. By now, congestion game theory is a well-known framework to study scenarios in which selfish agents strive to allocate resources. The following example is a typical problem addressed using the congestion game theory. Suppose we have a finite set of agents and a finite set of roads. Each agent has a starting point and a destination. Each agent strives to choose a set of roads such that the agent's total latency (travelling time) is minimized. The latency of each road is a function of the congestion of that road (number of agents choosing that road). Standard congestion game theory assumes each agent has the same effect on a road's congestion while weighted congestion game theory assumes agents have different effects on a road's (resource's) congestion.

1.3 Contributions

In this thesis, we investigate MRTA problems for both homogeneous and heterogeneous robot teams. For both cases, we consider two types of resource allocation problems, with increasing marginal gain or decreasing marginal gain. We discuss the suboptimality of the approximate solutions and conduct simulations and experiments on Robotarium [7].

For MRTA problems for homogeneous robot teams, our contributions lie in the following aspects. (1) We introduce standard congestion games to design decentralized MRTA algorithms to give approximate solutions to resource allocation problems. (2) We unify the decentralized best response dynamics proposed in [8] under the framework of standard congestion games, and we show that the assumption of robots' monotonically decreasing payoff functions is not necessary. (3) We present a concurrent version of best response dynamics. Based on the technique developed in [9], we specifically tailor the parameters and show convergence to a pure Nash equilibrium. (4) We present a decentralized dual greedy algorithm and show its convergence to a pure Nash equilibrium. (5) We discuss the suboptimality lower bound of a pure Nash equilibrium using $\lambda - \mu$ smoothness techniques [10]. (6) Simulations are conducted to verify the scalability of the above algorithms. We test suboptimality of a pure Nash equilibrium in simulations. (7) We implement the above algorithms using real robots in the Robotarium [7].

For MRTA problems for heterogeneous robot teams, our contributions come from the following aspects. (1) We introduce weighted congestion games to design decentralized MRTA algorithms to provide approximate solutions to resource allocation problems. (2) We prove the decentralized best response dynamics converge to a pure Nash equilibrium. (3) We discuss the suboptimality lower bound of a pure Nash equilibrium using $\lambda - \mu$ smoothness techniques [10]. (4) Simulations are conducted to verify the scalability and the suboptimality of a pure Nash equilibrium. (5) We implement decentralized best response dynamics for weighted congestion games using real robots in the Robotarium [7].

1.4 Thesis Overview

The remainder of the thesis is outlined as follows. In chapter 2, relevant studies on MRTA and congestion games are reviewed. In chapter 3, the standard congestion game theory is exploited to provide approximate solutions to the resource allocation problems for homogeneous robot teams. In chapter 4, the weighted congestion game theory is explored

to solve the resource allocation among the heterogeneous robot teams. In chapter 5, the findings are summarized and future research directions are discussed.

CHAPTER 2

RELATED WORK

2.1 Multi-Robot Task Allocation

The task allocation problem is fundamental in the field of multi-robot systems research since robot teams need to allocate tasks to team members before performing the tasks. Gerkey and Matarić [11] categorize task allocation problems according to the characteristics of robots and tasks. Robots are single-task (ST) robots if they can only execute a single task at a time or multi-task (MT) robots if they can perform multiple tasks concurrently. Multi-robot (MR) tasks require more than one robot to complete while single-robot (SR) tasks require only one robot. Task allocation problems can also be categorized according to the planning horizon: instantaneous assignment (IA) or time-extended assignment (TA). IA means robot teams only need to allocate tasks once based on current information. TA indicates that robot teams need to allocate tasks dynamically based on new information. We limit the scope to IA type problems in this thesis.

Centralized approaches for solving task allocation problems exist in the literature. ST-SR problems are the simplest among all four categories, and these can be solved in polynomial time [12]. The goal is to find a one-to-one matching between robots and tasks such that the sum of utilities of all robots is maximized. ST-SR problems can be formulated as finding a perfect weighted matching on a bipartite graph, and hence can be solved by Hungarian methods [13]. The ST-MR problem, known as the coalition formation problem in the operations research community, can be modeled as a set partitioning problem [11]. In the context of multi-robot research, set partition problems, which are NP-hard, aim to allocate robots to disjoint coalitions in order to maximize overall utility. Exact solutions and heuristics for set partitioning problems are well studied [14, 15]. Readers are referred to

Rasmussen [16] for a comprehensive review of exact and heuristic solutions. For the MT-SR problem, Gerkey and Matarić have shown it is mathematically equivalent to the ST-MR problem [11]. MT-MR problems can be formulated as set covering problems, which are also well studied [17, 18, 19]. Set covering problems aim to assign robots to (possibly overlapping) coalitions such that the sum of overall utilities is maximized. Readers are referred to Caprara [20] for a survey of both exact and heuristic solutions.

All types mentioned above of task allocation problems can be formulated as optimization problems for which the objective function is the sum of robots' utilities. The unique advantage of centralized approaches is that they provide good solution quality (if not optimal) for the optimization problem. However, centralized approaches rely on a central agent. Hence, centralized approaches are not robust to the failure of the central agent and suffer from scalability issues [5].

Numerous efforts have been made towards decentralizing robot task allocation algorithms. Among all these efforts, market-based or auction methods, which typically have both centralized and decentralized components, are the most popular. Many real-world multi-robot task allocation systems are based on market-based approaches (Traderbots [21], Murdoch [22], Alliance [23]). Market-based approaches involve auction processes, during which robots bid for tasks based on their utilities for different tasks. For the ST-SR problem, market-based approaches can be completely decentralized by using consensus algorithms[24, 25, 26] with a lower bound on solution quality [27]. Luo and Sycara [28] propose a decentralized auction algorithm with a lower bound on solution quality when dependent tasks form groups. For the MT-SR problem, Lin and Zheng [29] develop a combinatorial bid mechanism. A manager robot of a task is responsible for selecting a subset of bidding robots based on the combination of their bids, optimally, according to different criteria. For the MT-MR problem, Vig and Adams [30] adapt multi-agent coalition formation algorithms [31], which are not market-based approaches, to the field of multi-robot task allocation. The coalition formation algorithm has two stages. Robots distributedly evaluate

and select a task and coalition pair in the first stage while achieving consensus on forming coalitions in the second stage. The algorithm addresses real robotics issues like failures of robots and communication loss between robots.

When multi-robot task allocation problems involve optimizing the sum of robots' utilities (when a robot's utility is fixed for a specific task) or the sum of robots' total traveling distance, there exist decentralized approaches providing approximate solutions [5]. However, there is a gap between centralized and decentralized approaches when the optimization problem being solved is a resource allocation problem. Resource allocation problems have many applications in fields of communication [32, 33], computational grids [34, 35] and cloud computing [36, 37]. One of the most popular resource optimization models is the multiple knapsack problem [38]. In the field of communication, radio access allocation problems are studied to minimize interference. In the field of computational grids or cloud computing, resource allocation problems are useful to make the full use of computational resources. In the context of multi-robot task allocation, resource allocation problems view robots as resources and allocate them to different tasks such that the overall performance is optimized. In [39], the ST-MR problem is considered as a resource allocation problem and solved by a centralized task coordinator.

One way to develop decentralized approaches is to let robots' utilities become functions of resources (number of robots) allocated to a task. Hence, the congestion game theory serves as a suitable tool since robots' utilities are functions of the number of robots participating in a task. In [8], hedonic game theory is used to solve ST-MR problems, although the authors of [8] do not apply their work to solve resource allocation problems. Hedonic games are the same as congestion games in the context of ST-MR problems, since robots have singleton strategy spaces. Weighted congestion game theory is a perfect tool to design decentralized approximation algorithms for resource allocation problems involving heterogeneous robot teams, as different robots are associated with different weights according to their impacts on different tasks. [40].

2.2 Congestion Games

First introduced by Rosenthal [6], congestion game theory captures resource sharing scenarios in which selfish players choose a subset of resources to maximize (minimize) their payoff (delay). A player's payoff for a resource is a function of the weights (load) using that resource. In this thesis, we consider two types of congestion games, namely standard congestion games and weighted congestion games. Standard congestion games associate the same weight to all agents, while weighted congestion games associate different weights to different agents.

Standard congestion games are potential games [40]. A potential game has a potential function, which could express the incentives of players to change their strategies. An advantage of the potential game is the existence of pure Nash equilibrium, which is proven by constructing a potential function [41]. If a potential function could be constructed for a game, then players' selfish behaviors, which increase their payoffs, would also strictly increase the potential function. Hence, for finite congestion games, a pure Nash equilibrium always exists. Sequential best response dynamics, during which players behave greedily, are guaranteed to reach pure Nash equilibria in congestion games thanks to potential functions.

In addition to sequential dynamics, concurrent dynamics have also proposed for congestion games [9]. Weighted congestion game theory is a generalization of the congestion game theory, as different players have different weights for different resources [42]. Unfortunately, weighted congestion games are not potential games in general [43]. Nevertheless, singleton weighted congestion games, in which players are only allowed to select only one resource, convergence to a pure Nash equilibrium is guaranteed if the payoff is monotonically decreasing or increasing with respect to the number of players in maximization or minimization games [43].

The price of anarchy (POA), which is an important concept for game theory, gives

a lower bound for the worst-case pure Nash equilibrium in terms of the sum of players' payoffs to the optimal value of the sum of players' payoffs [44]. Much work has been done to discuss the POA for congestion games [45, 44, 46]. Among all these works, $\lambda - \mu$ smoothness arguments provide a unified framework to derive the POA for different utility functions [45]. The $StrongNashequilibrium$ is another solution concept in game theory, which allows players to form groups and take coordinated actions [47]. The set of strong Nash equilibrium is a subset of the set of pure Nash equilibrium since every strong Nash equilibrium is a pure Nash equilibrium but not vice versa. Since the set of strong Nash equilibrium is a subset of the pure Nash equilibrium, it is possible to obtain better POA bounds for some utility functions in congestion games [48, 49].

Congestion games have many applications in the field of communications, autonomous driving, and power grids [50, 51]. In the field of communications, congestion game theory is used to select distributed broadband network to minimize radio access interference [50]. In autonomous driving, altruistic autonomous cars can be designed to sacrifice their own interests and to influence human drivers to lead to better POA bounds under the framework of congestion game theory and finally decrease traffic congestion [51].

CHAPTER 3
STANDARD CONGESTION GAME-BASED TASK ALLOCATION FOR HOMOGENEOUS ROBOT TEAMS

In this chapter, we illustrate how to utilize standard congestion game theory to solve task allocation problems for homogeneous robot teams. We model the task allocation problem for homogeneous robot teams as two different types of resource allocation problems. We explain the decentralized sequential best response dynamics algorithm developed in [8] and prove the convergence to a pure Nash Equilibrium under the framework of standard congestion game theory. By using standard congestion game theory, we show that the assumption of robots' monotonically decreasing utility function in [8] is not necessary. A decentralized concurrent best response dynamics algorithm is developed. We also present a decentralized dual greedy algorithm and show its convergence to a pure Nash equilibrium. Lower bounds of suboptimality for pure Nash equilibria with respect to the resource allocation problems are discussed. Numerical simulations and experiments using real robots in the Robotarium [7] are conducted to verify the effectiveness of the above algorithms.

3.1 Preliminaries

In this section, we give the definitions of the standard congestion game, pure Nash equilibrium, strong Nash equilibrium, and potential function in the context of the ST-MR task allocation problem.

Definition 1. *A standard congestion game model is a tuple* $(N, T, (\Sigma_i)_{i\in N}, (\overline{l_j})_{j\in T}, (P_i)_{i\in N})$*, where (1)* $N = \{1, 2, ..., n\}$ *denotes the set of* n *robots; (2)* $T = \{1, 2, ..., t\}$ *is the set of* t *different tasks; (3)* $\Sigma_i \subseteq T$ *is the strategy space for robot i, which contains all tasks robot i could choose; (4)* $\overline{l_j}$ *is the latency function for task j, and the latency of a task measures*

the congestion of that task, which depends on how many robots are assigned to that task; (5) P_i is the payoff function for robot i.

Denote $\boldsymbol{S} = \Sigma_1 \times \Sigma_2 ... \times \Sigma_N$. A strategy profile of the standard congestion game is a vector $\boldsymbol{s} = (s_1, ..., s_n) \in \boldsymbol{S}$, where robot $i \in N$ chooses a task $s_i \in \Sigma_i$. The latency function $\overline{l_j(\boldsymbol{s})}$ for task j is a function mapping from $\boldsymbol{S}$ to $\mathbb{R}$. Given a strategy profile s, let n_j denote the number of robots assigned to task j. The latency function $\overline{l_j(\boldsymbol{s})}$ for task j only depends on n_j such that $\overline{l_j(\boldsymbol{s})} = l_j(n_j)$.

Given a strategy profile $\boldsymbol{s}$, Robot i's payoff function $P_i(\boldsymbol{s})$ is equal to task s_i's latency function such that $P_i(\boldsymbol{s}) = \overline{l_{s_i}(\boldsymbol{s})} = l_{s_i}(n_{s_i})$, where s_i is the task robot i is assigned to in strategy profile $\boldsymbol{s}$. Let $J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s})$. For $J(\boldsymbol{s})$, we have the following relationship:

$$J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) = \sum_{j \in T} n_j \overline{l_j(\boldsymbol{s})} = \sum_{j \in T} n_j l_j(n_j)$$

We consider payoff maximization games, in which robots strive to maximize their own payoff.

Definition 2. *A strategy profile $\boldsymbol{s}$ is a pure Nash equilibrium if no robot could increase its payoff by unilaterally choosing a new task, that is, if the following inequality:*

$$P_i(\boldsymbol{s}) \geq P_i(\boldsymbol{s}^{'})$$

where $\boldsymbol{s}^{'} = (s_1, ..., s_i^{'}, s_{i+1}, ..., s_n)$, holds for $\forall i \in N$ and $\forall s_i^{'} \in \Sigma_i$.

Definition 3. *A strategy profile $\boldsymbol{s}$ is a strong Nash equilibrium if no coalitions of robots (robots form groups to choose tasks in a coordinated way) could change their tasks in a way such that every robot of the coalition could increase its payoff, i.e., for any strategy profile $\boldsymbol{s}^{'} \neq \boldsymbol{s}$, there at least exists a robot j such that $s_j \neq s_p$ and $P_j(\boldsymbol{s}) \geq P_j(\boldsymbol{s}^{'})$.*

Definition 4. *An exact potential function is a function Φ mapping from $\boldsymbol{S}$ to $\mathbb{R}$, such that*

for $\forall i \in N$ *and* $\forall \boldsymbol{s} = (s_1, ..., s_i, ..., s_n), \boldsymbol{s}^{'} = (s_1, ..., s_i^{'}, s_{i+1}, ..., s_n) \in \boldsymbol{S}$*,*

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}^{'}) = P_i(\boldsymbol{s}) - P_i(\boldsymbol{s}^{'}).$$

Put it in another way, if a robot unilaterally changes its task, the change of the exact potential function is equal to the change of the robot's payoff.

Definition 5. *A congestion game is an exact potential game if there exists an exact potential function.*

Assumption 1. *We assume the communication network for robots is fully connected, i.e., there is a communication link between each pair of robots.*

3.2 Two Resource Allocation Problems For Task Allocation For Homogeneous Robot Teams

Definition 6. *Suppose we have a task and its performance metric. The* marginal gain *of a task is defined as the improvement of the task's performance by adding one new participating robot to that task.*

In this section, we formulate the task allocation problem for homogeneous robot teams as two resource allocation problems. The first resource allocation problem assumes the better performance of the task with more participating robots. However, the corresponding marginal gain with respect to the number of participating robots is monotonically decreasing. There exist real-world robot tasks that align with this assumption. Consider a team of robots tracking a moving target and collaboratively construct an observability matrix. The trace of the observability matrix is a measure of the tracking performance, which is a submodular function [52]. A submodular function is a set function. Suppose we have a set X and set Y such that $Y \subset X$. A submodular function f has the property that $f(z \cup X) - f(X) < f(z \cup Y) - f(Y)$ for any z in the domain of f. Hence, the marginal

gain is monotonically decreasing. For exploration tasks that aim to maximize mutual information of sensor measurements, mutual information is a submodular function [53]. Again, the marginal gain for exploration tasks is monotonically decreasing.

The second resource allocation problem assumes the better performance of the task with more participating robots, and the corresponding marginal gain regarding the number of participating robots is monotonically increasing. This assumption could be true in real-world robotics applications when the number of participating robots is not very large. For example, when robots teaming for information gathering or transporting an object, the marginal gain of the performance of the task could be monotonically increasing with respect to the number of participating robots.

These two types of resource allocation problem are simplifications of real-world robotic applications, and we aim to use them to illustrate how to design decentralized algorithms via congestion game theory to give approximate solutions to resource allocation problems. More complicated real-world resource allocation problems could also be solved using congestion game theory, thanks to its hierarchical structure (see figure 3.1). For a given resource allocation problem, a global objective function needs to be optimized. We could decompose the global objective function such that the sum of all robots' payoff functions is the global function. Robots would selfishly optimize their payoff functions and reach a pure Nash equilibrium via congestion game. In return, the pure Nash equilibrium provides

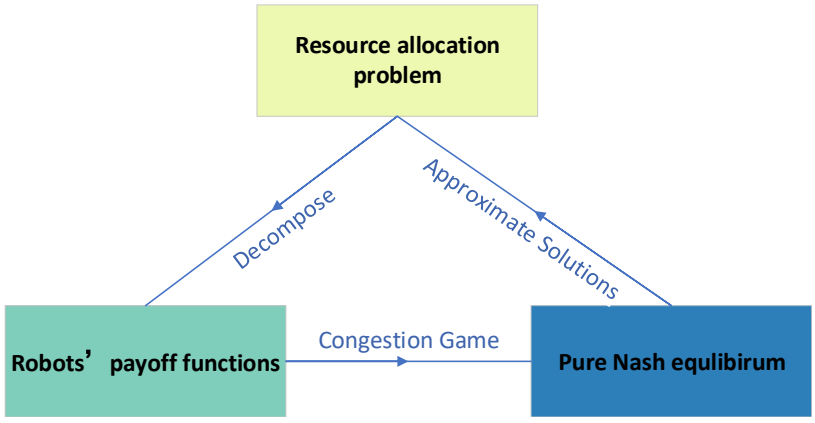


Figure 3.1: Utilizing the hierarchical structure of congestion game

an approximate solution to the resource allocation problem.

3.2.1 Marginal Gain Decreasing With Respect To the Number of Participating Robots

For the first type of resource allocation problem, with marginal gain monotonically decreasing with respect to the number of participating robots (see figure 3.2), the optimization problem is formulated as:

$$\max_{n_1,n_2,...,n_t} \quad \sum_{j=1}^{t} a_j \log(1+n_j) \tag{3.1}$$

where $n_1, n_2, ..., n_t$ is the number of robots participating in task $1, 2, ..., t$, $\sum_{j=1}^{t} n_j = n$, n is the total number of robots, and $a_j \in [0, 1]$ is the associated weight (priority) of task *j*.

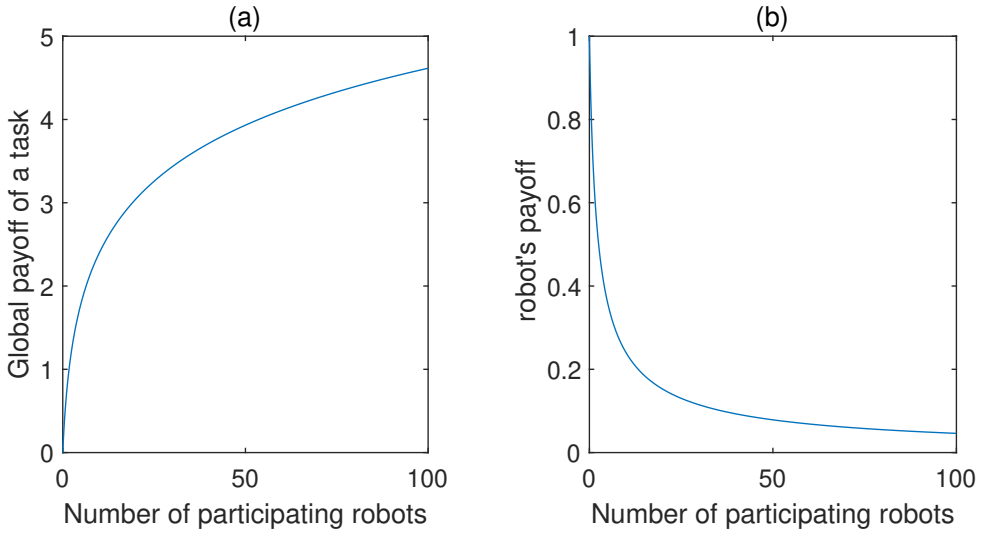


Figure 3.2: (a) global payoff of a task, i.e, the sum of all participating robots' payoff ;(b) robot's payoff function

Given a strategy profile $\boldsymbol{s}$, the corresponding robot level payoff function for robot *i* is:

$$P_i(\boldsymbol{s}) = \overline{l_{\boldsymbol{s}_i}(\boldsymbol{s})} = l_{\boldsymbol{s}_i}(n_{s_i}) = \begin{cases} \frac{a_{\boldsymbol{s}_i} \log(1+n_{\boldsymbol{s}_i})}{n_{\boldsymbol{s}_i}} & \text{if } n_{\boldsymbol{s}_i} > 0 \\ 0 & \text{if } n_{\boldsymbol{s}_i} = 0 \end{cases} \tag{3.2}$$

where s_i is the task robot *i* chooses in the strategy profile $\boldsymbol{s}$, $\overline{l_{\boldsymbol{s}_i}}$ is the latency function of task $\boldsymbol{s}_i$ and $n_{\boldsymbol{s}_i}$ is the number of robots participating in task s_i. The robot level payoff function $P_i(\boldsymbol{s})$ is monotonically decreasing with respect to the number of co-working robots robots assigned to task s_i. By summing over all robots' payoff functions, optimizing the sum of robots' payoff become equivalent to optimizing the resource allocation problem in equation (3.1). To be more specific, we have the following relationship:

$$J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) = \sum_{j \in T} n_j l_j(n_j) = \sum_{j=1}^{t} a_j \log(1 + n_j)$$

3.2.2 Marginal Gain Increasing With Respect to the Number of Participating Robots

For the second type of resource allocation problem, with marginal gain monotonically decreasing with respect to the number of participating robots (see figure 3.3), the optimization problem can be formulated as:

$$\begin{aligned} \max_{n_1, n_2, ..., n_t} \quad & \sum_{j=1}^{t} a_j n_j^2 \\ \text{s.t.} \quad & n_j \leq c_j, \quad j = \{1, 2, ..., t\} \end{aligned} \tag{3.3}$$

where $n_1, n_2, ..., n_t$ is the number of robots participating in task $1, 2, ..., t$, $\sum_{j=1}^{t} n_j = n$, n is the total number of robots, a_j is the associated weight (priority) of task *j*, c_j is the maximum number of participating robots for task *j* and $\sum_{j=1}^{t} c_j \geq n$. There are two reasons for having a constraint on the maximum number of participating robots for a task: (1) If there are no constraints, all robots would simply choose the task with the largest value for a_j, leaving rest of the tasks unassigned; (2) There could be physical limitations which lead to a constraint on the maximum number of participating robots for the individual tasks.

Remark 1. *This is a simple optimization problem. However, its counterpart (see section 4.2.1) in weighted congestion games is equivalent to the multiple quadratic knapsack prob-*

lem, which is NP-hard [38].

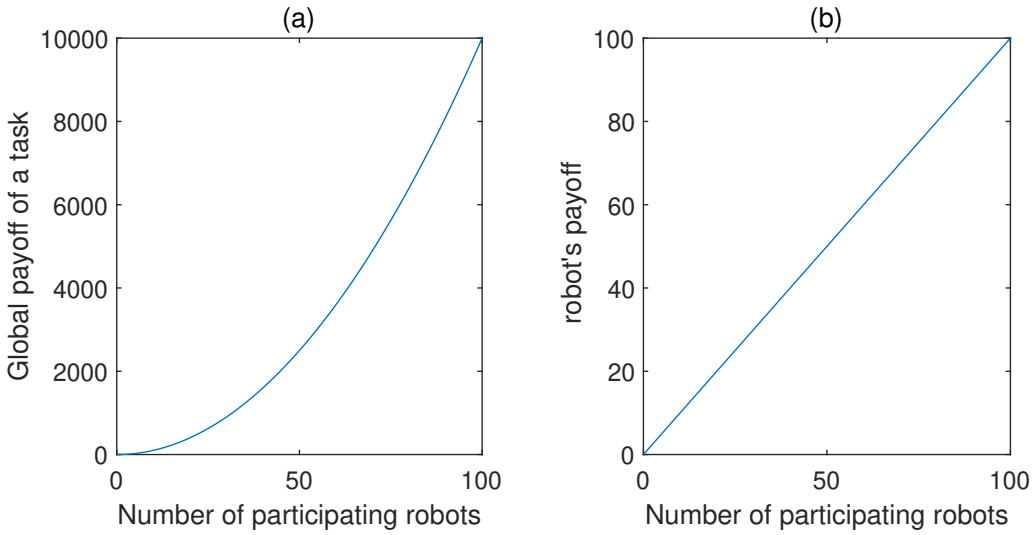


Figure 3.3: (a) global payoff of a task, i.e, the sum of all participating robots' payoff ;(b) robot's payoff function when the number of co-working robots satisfies the constraint.

Given a strategy profile $\boldsymbol{s}$, the corresponding robot level payoff function for robot i is:

$$P_i(\boldsymbol{s}) = \overline{l_{s_i}(\boldsymbol{s})} = l_{\boldsymbol{s}_i}(n_{s_i}) = \begin{cases} a_{s_i} n_{\boldsymbol{s}_i} & \text{if } n_{s_i} \leq c_{s_i} \\ 0 & \text{if } n_{s_i} > c_{s_i} \end{cases} \tag{3.4}$$

where s_i is the task robot *i* chooses in the strategy profile $\boldsymbol{s}$, $\overline{l_{s_i}}$ is the latency function of task s_i and n_{s_i} is the number of robots participating in task s_i. The robot level payoff function $P_i(\boldsymbol{s})$ is monotonically increasing with respect to the number of co-working robots when the number of co-working robots is no more than c_{s_i}. If the number of co-working robots is greater than c_{s_i}, robot i's payoff becomes zero. This ensures the numbers of participating robots for all tasks satisfy the constraints in any resultant pure Nash equilibrium. By summing over all robots' payoff functions, optimizing the sum of robots' payoff becomes equivalent to optimizing the the resource allocation problem defined by (3.3). To be more

specific, we have the following relationship:

$$J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) = \sum_{j \in T} n_j l_j(n_j) = \sum_{j=1}^{t} a_j n_j^2$$

.

3.3 Decentralized Sequential Best Response Dynamic

In this section, we unify the decentralized sequential best response dynamics developed in [8] under the framework of congestion game theory. By using congestion game theory, we show that the monotonically decreasing assumption of robots' payoff functions in [25] is unnecessary for convergence. Hence, decentralized sequential best response dynamic work for both resource allocation problems in section 3.2.1 and section 3.2.2. We also provide time complexity analysis from the perspective of congestion game theory.

3.3.1 Algorithm Description

Robot *i* has four variables, namely *satisfied*, $round^i$, $randnum^i$ and Π^i. If *satisfied* is true, robot *i*'s current task maximizes its payoff . Hence, robot *i* has no incentive for choosing a new task. If *satisfied* is false, robot *i* could choose the task to achieve the maximum payoff. $\Pi^i(m)$ is robot *i*'s data for which task robot *m* is choosing. Π^i is the set of $\Pi^i(m), \forall m \in N$, which contains data about all robots' current chosen tasks. $round^i$ indicates how many rounds Π^i has evolved. Each time Π^i is updated, $round^i$ is incremented by one. $randnum^i$ is a random number, which is assigned a random value whenever Π^i is updated. M^i is the collection of robot *i*'s three variables, $\Pi^i, round^i$ and s^i.

For the initialization phase (lines 1-8), robot *i* randomly chooses a task (line 3). Robot *i*'s $\Pi^i(i)$ is updated accordingly (line 4). Then robot *i* broadcasts its own M^i and receives $M^k, \forall k \neq i \in N$, from other robots (line 5). Robot *i* updates Π^i based on data received subsequently (line 6-8).

Algorithm 1: Decentralized sequential best response dynamics for each robot $i \in$ N

/* Initialization */

satisfied←false; $round^i \leftarrow 1$; $randnum^i \leftarrow 0$;

$\Pi^i = \{\Pi^i(m) \leftarrow 0, \forall m \in N\}$;

Randomly choose a task $j \in$ T;

Update $\Pi^i(i) \leftarrow j$;

Broadcast $M^i = \{\Pi^i, \quad round^i, \quad randnum^i\}$ and receive $M^k = \{\Pi^k, \quad round^k, \quad randnum^k\}, \forall k \neq i \in N$;

for *every* $k \neq i \in N$ **do**

  $\Pi^i(k) \leftarrow \Pi^k(k)$;

end

/* Main loop for sequential best response dynamics */

while *true* **do**

  if *satisfied* == *false* **then**

    choose task j^* which maximizes robot i's payoff;

    if $j^* \neq \Pi^i(i)$ **then**

      $round^i \leftarrow round^i + 1$;

      $randnum^i \leftarrow$ a random number drawn from unif[0,1];

      $\Pi^i(i) \leftarrow j^*$;

    end

    satisfied ← true;

  end

  Broadcast $M^i = \{\Pi^i, \quad round^i, \quad randnum^i\}$ and receive $M^k = \{\Pi^k, \quad round^k, \quad randnum^k\}, \forall k \neq i \in N$;

  construct the set $K^\star \leftarrow \{k \neq i \in N \mid \arg\max_k \quad round^k\}$;

  $k^{\star\star} \leftarrow \arg\max_{k \in K^\star} \quad randnum^k$;

  if $(round^i < round^{k^{\star\star}}) \quad or \quad (round^i == round^{k^{\star\star}} \quad \& \quad randnum^i < randnum^{k^{\star\star}})$ **then**

    $\Pi^i \leftarrow \Pi^{k^{\star\star}}$; $round^i \leftarrow round^{k^{\star\star}}$; $randnum^i \leftarrow randnum^{k^{\star\star}}$;

    satisfied ← false;

  end

end

After initialization, robot *i* starts to execute the main loop of sequential best response dynamics (lines 9-26). If robot i's *satisfied* equals false, then robot i would choose a task j^* which maximizes its payoff (lines 10-11). If task j^* is not the same as $\Pi^i(i)$, then $\Pi^i(i), round^i$ and $randnum^i$ are updated accordingly (lines 12-16). Robot i's satisfied is set to true (line 17). Next, robot i broadcasts its own M^i and receives $M^k, \forall k \neq i \in N$, from other robots (lines 19). Based on the data received, robot i first constructs set $K^\star$, such that each robot $k \in K^\star$'s $round^k$ is the largest in the set N (line 20). Then robot i finds robot $k^{\star\star}$, whose $randnum^{k^{\star\star}}$ is the largest in the set $K^\star$ (line 21). If i is not equal to $k^{\star\star}$, robot i's $\Pi^i, round^i$, $randnum^i$ and ***satisfied*** are assigned the values of $\Pi^{k^{\star\star}}, round^{k^{\star\star}}$, $randnum^{k^{\star\star}}$ and false (lines 22-25). Afterward, robot i proceeds to execute the above while loop again.

It is important to note that lines 19-25 ensure only one robot could select a new task during each iteration of the while loop. Also, robots behave greedily to choose tasks that maximize their own payoffs. Hence, this algorithm is called decentralized sequential best response dynamic.

3.3.2 Convergence to a Pure Nash Equilibrium

Theorem 1. *Decentralized sequential best response dynamics converge to a pure Nash equilibrium.*

Proof. For standard congestion games, an exact potential function $\Phi(\boldsymbol{s})$ could be constructed [6]:

$$\Phi(\boldsymbol{s}) = \sum_{p \in T} \sum_{m=1}^{n_p} l_p(m) \tag{3.5}$$

where n_p is the number of robots choose task p in strategy profile $\boldsymbol{s}$. To be more specific, when robot i switches from $s_i = p$ to $s_i^{'} = q$, the change of the exact potential function is:

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}^{'}) = l_p(n_p) - l_q(n_q + 1) \tag{3.6}$$

where $\boldsymbol{s}^{'} = (s_1, ..., s_i^{'} = q, s_{i+1}, ..., s_n)$, n_q is the number of robots that choose task q in strategy profile $\boldsymbol{s}$. By definition, the above change is equivalent to the change of robot i's payoff:

$$l_p(n_p) - l_q(n_q + 1) = \overline{l_p(\boldsymbol{s})} - \overline{l_q(\boldsymbol{s}^{'})} = P_i(\boldsymbol{s}) - P_i(\boldsymbol{s}^{'}) \tag{3.7}$$

During each iteration of sequential best response dynamics, a robot behaves greedily to maximize its payoff. Every time a robot switches to a new task, the change of its payoff function is strictly positive. Hence, the exact potential function is strictly increasing during the process of sequential best response dynamics. Because we consider a finite number of robots and tasks, sequential best response dynamics would converge to a pure Nash equilibrium, in which no robot could unilaterally increase its payoff. □

Since the exact potential function for standard congestion games has no assumptions of robots' payoff functions or tasks' latency functions, there is no need to assume monotonically decreasing payoff functions.

3.3.3 Time Complexity Analysis

We refer to the unit of time required for each robot to execute an iteration of the sequential best response dynamics (lines 10-25) as a time step. A robot would execute at most nt^2 iterations of the sequential best response dynamics before converging to a pure Nash equilibrium [54], since the potential function has at most nt^2 distinct values. A robot needs to investigate t tasks in an iteration (line 11). The complexity for constructing the set $K^{\star}$ and finding $k^{\star\star}$ is $O(n)$ (lines 20-21). Hence, the time complexity is $O(n^2t^3)$.

3.4 Decentralized Concurrent Best Response Dynamics

In this section, we present a concurrent version of best response dynamics. Based on the technique developed in [9], we specifically tailor the parameters for the resource allocation problem in section 3.2.1 and show convergence to a pure Nash equilibrium. Concurrent

dynamics allow all robots to make a decision per iteration, while only a single robot could make a decision per iteration in sequential dynamics. Hence, concurrent best response dynamics require less coordination among robots than sequential best response dynamics.

In previous sections, we consider payoff maximization games. For the convenience of proof, in this section we consider the equivalent payoff minimization games by taking the opposite of robots' payoff functions in maximization games.

3.4.1 Algorithm Description

Algorithm 2: Decentralized concurrent best response dynamics for each robot $i \in$ N

```
   /* Initialization                                              */
1  Π^i = {Π^i(m) ← 0, ∀m ∈ N};
2  Randomly chooses a task j ∈ T;
3  Π^i(i) ← j;
4  Broadcast Π^i and receive Π^k, ∀k ≠ i ∈ N;
5  for every k ≠ i ∈ N do
6  |  Π^i(k) ← Π^k(k);
7  end
   /* Main loop for concurrent best response dynamics             */
8  while true do
9  |  j* ← concurrent protocol(Π^i);
10 |  if j* ≠ Π^i(i) then
11 |  |  Π^i(i) ← j*;
12 |  end
13 |  Broadcast Π^i and receive Π^k, ∀k ≠ i ∈ N;
14 |  for every k ≠ i ∈ N do
15 |  |  Π^i(k) ← Π^k(k);
16 |  end
17 end
```

Robot i has one variable Π^i. $\Pi^i(m)$ is robot i's data for which task robot m is choosing. Π^i is the set of $\Pi^i(m), \forall m \in N$, which contains robot i's data about all robots' current chosen tasks.

For the initialization phase (lines 1-7), robot *i* randomly chooses a task (line 2). Robot *i*'s $\Pi^i(i)$ is updated accordingly (line 3). Then robot *i* broadcasts its own Π^i and receives

Algorithm 3: Concurrent protocol for robot $i \in$ N

1 **Function** *concurrent protocol* (Π^i)

/* let $\boldsymbol{s}$ denote the strategy profile encoded in Π^i */

2 $p \leftarrow \Pi^i(i)$;

3 Let robot i sample a task $j \in$ T uniformly at random;

4 **if** $l_{\Pi^i(i)}(n_{\Pi^i(i)}) > l_j(n_j + 1)$ **then**

5 $p \leftarrow j$ with migration probability

$$\min\{1, 0.25\frac{tl_{max}}{n}\frac{l_{\Pi^i(i)}(n_{\Pi^i(i)}) - l_j(n_j+1)}{l_{\Pi^i(i)}(n_{\Pi^i(i)})}\}$$

6 **end**

7 return p;

$\Pi^k, \forall k \neq i \in N$, from other robots (line 4). Robot *i* updates Π^i based on data received subsequently (line 5-7).

For the main loop of concurrent best response dynamics (lines 8-17), robot i first executes *concurrent protocol* (line 9). During the *concurrent protocol* (algorithm 3), robot i samples a task j uniformly at random (line 3 in algorithm 3). Next, n_j and $n_{\Pi^i(i)}$ are the numbers of robots doing task j and task $\Pi^i(i)$, which could be obtained via Π^i. If $l_j(n_j+1)$ is smaller than $l_{\Pi^i(i)}(n_{\Pi^i(i)})$, then p would be assigned the value of j with the probability specified in line 5 of algorithm 3, where $l_{max} = \max_{j \in T} l_j(n)$. Afterward, the concurrent protocol is completed and $j^\star$ is assigned the value of p (line 9 in algorithm 2).

Next, if $\Pi^i(i)$ is not equal to $j^\star$, $\Pi^i(i)$ is assigned the value of $j^\star$ (lines 10-12). Robot i broadcasts Π^i and receives Π^k from other robots (line 13). Π^i is updated according to the data received (line 14-16). Afterward, robot i proceeds to execute the above while loop again.

Remark 2. *(Explanation of the probability in algorithm 3): In algorithm 3, p is assigned the value of j with probability of $\min\{1, 0.25\frac{tl_{max}}{n}\frac{l_{\Pi^i(i)}(n_{\Pi^i(i)})-l_j(n_j+1)}{l_{\Pi^i(i)}(n_{\Pi^i(i)})}\}$, where t is the number of tasks, $l_{max} = \max_{j \in T} l_j(n)$, n is the number of robots, n_j and $n_{\Pi^i(i)}$ are the numbers of robots doing task j and task $\Pi^i(i)$, $l_{\Pi^i(i)}$ is the latency function of task $\Pi^i(i)$ and l_j is the latency function of task j. This probability ensures decentralized concurrent best response*

dynamics would converge to a pure Nash equilibrium (see the proof of theorem 2).

3.4.2 Convergence to a Pure Nash Equilibrium

Assumption 2. *For payoff maximization (minimization) games, Decentralized concurrent best response dynamics assume a task's latency function is monotonically decreasing (increasing) with respect to the number of participating robots.*

This assumption ensures that concurrent best response dynamics converge to a pure Nash equilibrium. Given a strategy profile, a robot's payoff is equivalent to the latency of this robot's assigned task. Hence, this assumption implies that concurrent best response dynamics are only applicable to resource allocation problems with decreasing marginal gains as discussed in section 3.2.1 but not for the resource allocation problem discussed in section 3.2.2. For resource allocation problems discussed in section 3.2.2, sequential best response dynamics remains applicable.

Theorem 2. *For resource allocation problems formulated in section 3.2.1, decentralized concurrent best response dynamics converge to a pure Nash equilibrium.*

Proof. For standard congestion games, an exact potential function $\Phi(\boldsymbol{s})$ could be constructed [6]:

$$\Phi(\boldsymbol{s}) = \sum_{j \in T} \sum_{m=1}^{n_j} l_j(m) \tag{3.8}$$

where n_j is the number of robots choose task j in strategy profile $\boldsymbol{s}$.

Given a strategy profile $\boldsymbol{s}$, let $\boldsymbol{s}^{'}$ denote the strategy profile after one round of concurrent dynamics (lines 9-16 in algorithm 2). Let the difference of $\Phi(\boldsymbol{s}^{'})$ and $\Phi(\boldsymbol{s})$ be:

$$\Delta\Phi(\boldsymbol{s}^{'}, \boldsymbol{s}) = \Phi(\boldsymbol{s}^{'}) - \Phi(\boldsymbol{s}) \tag{3.9}$$

Define x_{jp} as the number of robots switching from task j to task p in one round of concurrent dynamics. In one round of concurrent dynamics, each robot makes a decision as

each of them was the only agent to make a decision. Hence, for robots switching from task j to task p, their payoffs in strategy profile s before their moves are $P_i(\boldsymbol{s}) = \overline{l_j(\boldsymbol{s})} = l_j(n_j)$, where n_j is the number of robots doing task j in strategy profile s. These robots believe their payoffs are $l_p(n_p+1)$ as each of them was the only agent to make a decision, where n_p is the number of robots doing task p in $\boldsymbol{s}$. The payoff is $l_p(n_p+1)$ since that robots believe that all robots are not making a decision except for themselves. However this is not true, as robots all make a decision in a round of concurrent dynamics. Since robots are minimizing their payoffs, switching from task j to p implies that $l_p(n_p+1) - l_j(n_j) < 0$. Otherwise, robots would not make a decision to switch from j to $j^{`}$.

We define the virtual potential gain $V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'})$ as:

$$V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'}) = x_{jp}(l_p(n_p+1) - l_j(n_j)) \tag{3.10}$$

where n_j and n_p are the number of robots doing task j and p in $\boldsymbol{s}$. $V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'})$ is the sum of the payoff changes each robots switching from task j to task p would contribute to the change of the potential function Φ if each of them was the only migrating agent.

In order to compensate for the fact that each agent makes a decision concurrently, we define the error term $F_j(\boldsymbol{s}, \boldsymbol{s}^{'})$ for a task j as:

$$F_j(\boldsymbol{s}, \boldsymbol{s}^{'}) = \begin{cases} \sum_{u=n_j+1}^{n_j+\Delta x_j} l_j(u) - l_j(n_j+1) & \text{if } \Delta x_j > 0 \\ \sum_{u=n_j+\Delta x_j+1}^{n_j} l_j(n_j) - l_j(u) & \text{if } \Delta x_j < 0 \\ 0 & \text{if } \Delta x_j = 0 \end{cases} \tag{3.11}$$

where $\Delta x_j = \sum_{j^\star \in T} x_{j^\star j} - x_{jj^\star}$ is the difference between the number of robots doing task j in profile $\boldsymbol{s}$ and $\boldsymbol{s}^{'}$.

Also, the following relationship holds true [9]:

$$\Delta\Phi(\boldsymbol{s}^{'}, \boldsymbol{s}) \leq \sum_{j,p \in T} V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'}) + \sum_{j \in T} F_j(\boldsymbol{s}, \boldsymbol{s}^{'}) \tag{3.12}$$

$V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'})$ is strictly negative since we are considering payoff minimization games. Next we need to show the sum of $F_j(\boldsymbol{s}, \boldsymbol{s}^{'})$ and $F_p(\boldsymbol{s}, \boldsymbol{s}^{'})$ is bounded by $-0.5V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'})$.

Suppose a robot i switches from task j to p in a round of concurrent dynamics. Robot i's contribution to $V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'})$ is $l_p(n_p + 1) - l_j(n_j)$. To bound its contribution to error terms $F_j(\boldsymbol{s}, \boldsymbol{s}^{'})$ and $F_p(\boldsymbol{s}, \boldsymbol{s}^{'})$, we consider the set $N^{'}$ of robots switching to p.

Recall that in the concurrent protocol (algorithm 3), robot i move to task p with it's migration probability $\min\{1, 0.25\frac{tl_{max}}{n}\frac{l_{\Pi^i(i)}(n_{\Pi^i(i)}) - l_j(n_j+1)}{l_{\Pi^i(i)}(n_{\Pi^i(i)})}\}$, where t is the number of tasks, $l_{max} = \max_{j \in T} l_j(n)$, n is the number of robots, n_j and $n_{\Pi^i(i)}$ are the numbers of robots doing task j and task $\Pi^i(i)$, $l_{\Pi^i(i)}$ is the latency function of task $\Pi^i(i)$ and l_j is the latency function of task j. We order set $N^{'}$ with robots' ascending migration probabilities. Let $\hat{X}_i$ denote the random number of robots ranking before robot i in $N^{'}$ according to the above ascending order. We upper bound robot i's contribution to $F_p(\boldsymbol{s}, \boldsymbol{s}^{'})$ as $l_p(n_p + \hat{X}_i) - l_p(n_p + 1)$.

Thanks to the migration probability defined in algorithm 3, we have:

$$\begin{aligned} E[l_p(n_p + \hat{X}_i) - l_p(n_p + 1)] &\leq E[n_p + \hat{X}_i - n_p] \\ &\leq n\frac{1}{t}0.25\frac{tl_{max}}{n}\frac{l_j(n_j) - l_p(n_p + 1)}{l_j(n_j)} \\ &\leq 0.25(l_j(n_j) - l_p(n_p + 1)) \end{aligned} \tag{3.13}$$

Recall we are considering equivalent payoff minimization games by taking the opposite of the payoff functions in section 3.2.1 (equation 3.2). The first inequality holds true as 1 is greater than the maximum slope of the payoff function. For the second inequality, it is true as there are at most n robots samples task p in a round of concurrent dynamics. The third inequality holds true since $\frac{l_{max}}{l_j} \leq 1$, as $l_{max} \geq l_j(n_j)$ and l_j, l_{max} are negative numbers (because we taking the opposite of the payoff functions in section 3.2.1).

We consider the set $N^{''}$ of robots departing from task j. We let $\bar{X}_i$ denote the random number of robots ranking before robot i in $N^{''}$ according to the ascending migration probabilities. We associate robot i's contribution to $F_j(\boldsymbol{s}, \boldsymbol{s}^{'})$ as $l_j(n_j) - l_j(n_j - \bar{X}_i + 1)$. Similarly, we have:

$$E[l_j(n_j) - l_j(n_j - \bar{X}_i + 1)] \leq 0.25(l_j(n_j) - l_p(n_p + 1)) \tag{3.14}$$

Combining inequalities (3.13) and (3.14), we have:

$$\begin{aligned} F_j(\boldsymbol{s}, \boldsymbol{s}^{'}) + F_p(\boldsymbol{s}, \boldsymbol{s}^{'}) &\leq 0.5(l_j(n_j) - l_p(n_p + 1)) \\ &= -0.5(l_p(n_p + 1) - l_j(n_j)) \end{aligned} \tag{3.15}$$

Thanks to inequality (3.15), we finally have:

$$\begin{aligned} E[\Delta\Phi(\boldsymbol{s}^{'}, \boldsymbol{s})] &\leq E[\sum_{j,p \in T} V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'}) + \sum_{j \in T} F_j(\boldsymbol{s}, \boldsymbol{s}^{'})] \\ &\leq 0.5E[\sum_{j,p \in T} V_{jp}(\boldsymbol{s}, \boldsymbol{s}^{'})] \\ &< 0 \end{aligned} \tag{3.16}$$

Hence, concurrent best response dynamics would converge to a pure Nash equilibrium in expectation. □

3.4.3 Time Complexity Analysis

We refer to the unit of time required for each robot to execute a round of the concurrent best response dynamics (lines 9-16) as a time step. Let $\boldsymbol{s}$ denote the strategy profile after initialization, $l_{max} = \max_{j \in T} l_j(n)$, $k = \min_{\boldsymbol{s}} \min_{j,p \in T, l_j(n_j) > l_p(n_p+1)} l_j(n_j) - l_p(n_p + 1)$. Decentralized concurrent best response dynamics would converge to a pure Nash equilibrium in expected time $O(\frac{\Phi(\boldsymbol{s}) n l_{max}}{l_{min} k^2})$ [9].

3.5 Decentralized dual greedy algorithm

In this section, we decentralize the dual greedy algorithm developed in [55] and show its convergence to a pure Nash equilibrium. For the convenience of the proof, we consider payoff equivalent payoff minimization games by taking the opposite of robots' payoff functions.

3.5.1 Algorithm Description

Each robot i has seven variables: U^i, Π^i, c^i, $nosol^i$, $j^i_\star$, $stop^i$ and $randnum^i$. U^i is the set of $\{u^i_1, u^i_2, ..., u^i_t\}$, where $u^i_1, u^i_2, ..., u^i_t$ indicates the constraint on maximum number of participating robots for each task. $\Pi^i(m)$ is robot i's data for which task robot m is choosing. Π^i is the set of $\Pi^i(m), \forall m \in N$. c^i is a variable which indicates whether robot i changes its strategy. $nosol^i$ is a variable which implies whether robot i could find a feasible solution. $stop^i$ indicates whether robot i sticks to its strategy and would not make any further changes. $j^i_\star$is a variable which indicates the task has maximum latency (line 10). $randnum^i$ is a random number, which is used for consensus.

For the initialization phase (lines 1-7), robot i randomly chooses a task and updates Π^i (line 3). Then robot i broadcasts Π^i and receives Π^k from other robots (line 4). Π^i is updated according to the data received (lines 5-7).

For the main loop of the decentralized dual greedy algorithm (lines 9-30), robot i first examines its variable $stop^i$ (line 9). If $stop^i$ equals to zero, $j^i_\star$ is set to be task j, which maximizes $l_j(u^i_j)$ (line 10). If $stop^i$ equals to one, $j^i_\star$ is set to zero (line 12). Next, robot i broadcasts $j^i_\star$ and receive $j^k_\star, \forall k \neq i \in N$ from other robots (line 14). $j^i_\star$ is updated according to the data received (line 15). $u^i_{j^i_\star}$ is decreased by one (line 16).

Afterward, robot i would execute dual greedy protocol (algorithm 5). If $\Pi^i \neq j^i_\star$, c^i, $nosol^i$, and $randnum^i$ are set to zero (lines 2-3 in algorithm 5). Let $npar(j^i_\star)$ denotes the number of robots doing task $j^i_\star$. If Π^i equals to $j^i_\star$ and $u^i_{j^i_\star} \geq npar(j^i_\star)$, c^i, $nosol^i$, and

Algorithm 4: Decentralized dual greedy algorithm for each robot $i \in$ N

/* Initialization */

$U^i = \{u_1^i \leftarrow n, u_2^i \leftarrow n, ..., u_t^i \leftarrow n\}, \quad c^i \leftarrow 0, \quad nosol^i \leftarrow 0, \quad j_\star^i \leftarrow 0, \quad stop^i \leftarrow 0, \quad randnum^i \leftarrow 0;$

$\Pi^i = \{\Pi^i(m) \leftarrow 0, \forall m \in N\};$

$\Pi^i(i) \leftarrow$ Randomly chooses a task $j \in$ T;

Broadcast Π^i and receive $\Pi^k, \forall k \neq i \in N;$

for *every* $k \neq i \in N$ **do**

  $\Pi^i(k) \leftarrow \Pi^k(k);$

end

/* Main loop for decentralized dual greedy algorithm */

while *max* $U^i > 0$ **do**

  if $stop^i == 0$ **then**

    $j_\star^i \leftarrow \max_{\{j \in T | u_j^i > 0\}} l_j(u_j^i);$

  else

    $j_\star^i \leftarrow 0;$

  end

  Broadcast $j_\star^i$ and receive $j_\star^k, \forall k \neq i \in N;$

  $j_\star^i \leftarrow \max_{k \in N} j_\star^k;$

  $u_{j_\star^i}^i \leftarrow u_{j_\star^i}^i - 1;$

  $c^i, \quad nosol^i, \quad randnum^i, \quad \Pi^i \leftarrow$ Dual greedy protocol $(\Pi^i, j_\star^i);$

  Broadcast $M^i = \{c^i, nosol^i, randnum^i, \Pi^i\}$ and receive $M^k, \forall k \neq i \in N;$

  $k^\star \leftarrow \max_{\{k \in N | c^k = 1\}} randnum^k;$

  if $nosol^{k^\star} == 0$ **then**

    if $c^i == 1$ **then**

      $\Pi^i(i) \leftarrow \Pi^{k^\star}(k^\star), c^i \leftarrow 0, \quad nosol^i \leftarrow 0, \quad randnum^i \leftarrow 0;$

    end

  else

    if $c^i == 1$ **then**

      $stop^i \leftarrow 1$

    else

      $u_{j_\star^i}^i \leftarrow 0, \quad c^i \leftarrow 0, \quad nosol^i \leftarrow 0, \quad randnum^i \leftarrow 0;$

    end

  end

end

Algorithm 5: Dual greedy protocol for robot $i \in$ N

1 **Function** *Dual greedy protocol* $(\Pi^i, j^i_\star)$
2 **if** $\Pi^i(i) \neq j^i_\star$ **then**
3 $c^i \leftarrow 0, \quad nosol^i \leftarrow 0, \quad randnum^i \leftarrow 0;$
4 **else**
  /* $npar(j^i_\star)$ is the number of robots doing task $j^i_\star$ based on the data of Π^i */
5 **if** $u^i_{j^i_\star} \geq npar(j^i_\star)$ **then**
6 $c^i \leftarrow 0, \quad nosol^i \leftarrow 0, \quad randnum^i \leftarrow 0;$
7 **else**
8 construct set $F = \{j \in T | u^i_j > npar(j)^i\}$;
9 **if** $F \neq \emptyset$ **then**
10 $\Pi^i(i) \leftarrow$ randomly chooses a task j from set F,
  $c^i \leftarrow 1, \quad nosol^i \leftarrow 0, \quad randnum^i \leftarrow unif[0,1]$;
11 **else**
12 $c^i \leftarrow 1, \quad nosol^i \leftarrow 1, \quad randnum^i \leftarrow unif[0,1]$;
13 **end**
14 **end**
15 **end**
16 return $c^i, \quad nosol^i, \quad randnum^i, \quad \Pi^i$;

$randnum^i$ are set to zero (line 6 in algorithm 5). If Π^i equals to $j^i_\star$ and $u^i_{j^i_\star} < npar(j^i_\star)$, robot i constructs a set F which contains all tasks such that $u^i_j > npar(j)^i$, e.g., the total number of participating robots for the task is less than the constraint on the number of maximim participating robots (line 8 in algorithm 5). If F is not an empty set, robot i randomly chooses a task from the set F (line 10 in algorithm 6). Next, c^i, $nosol^i$ are set to $\{1, 0\}$. Also, $randnum^i$ is assigned a random value draws uniformly from $[0, 1]$ (line 10 in algorithm 5). If F is an empty set, c^i, $nosol^i$ are set to $\{1, 1\}$. Also, $randnum^i$ is assigned a random value draws uniformly from $[0, 1]$ (line 12 in algorithm 5).

Robot i broadcasts $M^i = \{c^i, nosol^i, s^i, \Pi^i\}$ and receives M^k from other robots subsequently (line 18). $k^\star$ is set to maximize $s^k, \{k \in N | c^k = 1\}$ (line 19). If $nosol^{k^\star}$ equals to zero and c^i equals to 1, then Π^i, c^i, $nosol^i$, $randnum^i$ are set the values of $\{\Pi^{k^\star}(k^\star), 0, 0, 0\}$ (line 22). If $nosol^{k^\star}$ equals to 1 and c^i equals to 1, then $stop^i$ is set to 1 (line 26). If $nosol^{k^\star}$ equals to zero and c^i equals to 0, then $u^i_{j^i_\star}$, c^i, $nosol^i$, $randnum^i$ are set the values of

$\{0, 0, 0, 0\}$ (line 28).

3.5.2 Convergence to a Pure Nash Equilibrium

Assumption 3. *For payoff maximization (minimization) games, Decentralized dual greedy algorithm assumes a task's latency function is monotonically decreasing (increasing) with respect to the number of participating robots.*

Similar to section 3.4.2, this assumption ensures that decentralized dual greedy algorithm converges to a pure Nash equilibrium.

Theorem 3. *In the context of the ST-MR task allocation problem, the set of strong Nash equilibrium is equivalent to the set of pure Nash equilibrium for standard congestion games with monotone latency functions [56].*

Theorem 4. *For resource allocation problems formulated in section 3.2.1, decentralized dual greedy algorithm converges to a pure Nash equilibrium.*

Proof. Let $N_1, N_2, ..., N_k$ denote k different groups of robots. N_k denotes the kth group of robots , whose stop variable is set to one. Every time a group of robots' stop variable is set to one, indicates $\sum_{\{j \in T | u_j > 0\}} u_j < \sum_{\{j \in T | u_j > 0\}} n_j$, where n_j is the number of robots doing task j. The rest of the proof for converging to a strong Nash equilibrium is followed by [55]. By theorem 3, decentralized dual greedy algorithm converges to a pure Nash equilibrium. □

3.5.3 Time Complexity Analysis

We refer to the unit of time required for each robot to execute an iteration of the decentralized dual greedy algorithm (lines 9-30) as a time step. A robot would execute at most nt iterations of the main loop of the dual greedy algorithm before converging to a pure Nash equilibrium. In each iteration, a robot needs to investigate t different tasks (line 10). The complexities of finding $\max_{k \in N} j^k_\star$ (line 15) and $\max_{\{k \in N | c^k = 1\}} randnum^k$ (line 19) are both $O(n)$. In each iteration, a robot executes dual greedy protocol once. In each dual greedy protocol,

a robot needs to investigate t different tasks to construct the set F (line 8 in algorithm 3). Hence, the time complexity is $O(n^2t^2)$.

3.6 Suboptimality

Definition 7. *For a given pure Nash equilibrium $\boldsymbol{s}$, we define the suboptimality ratio α as:*

$$\alpha = \frac{J(\boldsymbol{s})}{J(\boldsymbol{s}^\star)}$$

where $J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s})$ is the sum of all robots' payoff, $J(\boldsymbol{s}^\star) = \sum_{i \in N} P_i(\boldsymbol{s}^\star)$ is the maximum value of the sum of all robots' payoff, and $\boldsymbol{s}^\star$ is the corresponding optimal strategy profile.

In this section, we investigate the lower bound of suboptimality for any resultant pure Nash equilibria.

3.6.1 Marginal Gain Increasing

We utilize the $\lambda - \mu$ smoothness technique developed in [10] to investigate lower bounds of suboptimality for the resource allocation problem formulated in section 3.2.2.

Assumption 4. *For a pure Nash equilibrium strategy profile $\boldsymbol{s}$, each task has at least one participating robots.*

This assumption is useful for further developments.

Suppose strategy profile $\boldsymbol{s}$ is a pure Nash equilibrium, and $\boldsymbol{s}^\star$ is a strategy profile that maximizes the sum of all robots' payoff. By the definition of pure Nash equilibrium, we have the following inequality for any robot i:

$$P_i(\boldsymbol{s}) \geq P_i(\boldsymbol{s}^{'}) \tag{3.17}$$

where $\boldsymbol{s} = (s_1, ..., s_i, ..., s_n)$, $\boldsymbol{s}^{'} = (s_1, ..., s_i^\star, ..., s_n)$ and $s_i^\star$ is the task robot i chooses in the optimal strategy profile $\boldsymbol{s}^\star$. For now, we assume after robot i moves from s_i to

$s_i^\star$, the resultant strategy profile still satisfies the constraint (e.g, the number of a task's participating robots does not exceed the constraint on the maximum number of participating robots).

Thanks to the hierarchical structure of congestion games, we could sum all robots' payoff:

$$\begin{aligned} J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) &\geq \sum_{i \in N} P_i(\boldsymbol{s}^{'}) \\ &\geq \sum_{\{j \in T | n_j^\star > 0\}} l_j(n_j) n_j^\star \end{aligned} \tag{3.18}$$

where n_j is the number of robots doing task j in profile $\boldsymbol{s}$ and $n_j^\star$ is the number of robots doing task j in strategy profile $\boldsymbol{s}^\star$. The first inequality holds true due to the definition of pure Nash equilibrium. There are two different cases for robots switching to task j. In the first case, robot i already chooses task j in strategy profile $\boldsymbol{s}$ and its payoff is $l_j(n_j)$ in strategy profile $\boldsymbol{s}^{'}$. In the second case, robot i chooses a task other than j in strategy profile $\boldsymbol{s}$. Its payoff is $l_j(n_j + 1)$ in strategy profile $\boldsymbol{s}^{'}$. The second inequality holds true since $l_j, \forall j \in T$ is monotonically increasing for this type of resource allocation problem.

Suppose there exists $\lambda > 0$ and $\mu > 0$ such that for arbitrary task j:

$$l_j(n_j) n_j^\star \geq \lambda l_j(n_j^\star) n_j^\star - \mu l_j(n_j) n_j, \quad n_j = \{1, \ldots, c_j - 1\}, \quad n_j^\star = \{1, \ldots, c_j\} \tag{3.19}$$

where c_j is the constraint on the maximum number of participating robots of task j. The reason for n_j ranging from 1 to $c_j - 1$ is that we assume after robot i moves from s_i to $s_i^\star$, the resultant strategy profile still satisfies the constraint. Assumption 4 n_j to start from 1 instead of 0.

Then we have:

$$\begin{aligned} J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) &\geq \sum_{\{j \in T | n_j^\star > 0\}} l_j(n_j) n_j^\star \\ &\geq \sum_{\{j \in T | n_j^\star > 0\}} \lambda l_j(n_j^\star) n_j^\star - \mu l_j(n_j) n_j \\ &\geq \sum_{\{j \in T\}} \lambda l_j(n_j^\star) n_j^\star - \mu l_j(n_j) n_j \\ &= \lambda J(\boldsymbol{s}^\star) - \mu J(\boldsymbol{s}) \end{aligned} \tag{3.20}$$

where $J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s})$ and $J(\boldsymbol{s}^\star) = \sum_{i \in N} P_i(\boldsymbol{s}^\star)$. The first inequality follows from (3.18). The second inequality follows from (3.19). The third inequality follows from $\mu > 0$.

Hence, we get a lower bound of suboptimality for any given pure Nash equilibrium $\boldsymbol{s}$:

$$\alpha = \frac{J(\boldsymbol{s})}{J(\boldsymbol{s}^\star)} \geq \frac{\lambda}{1+\mu} \tag{3.21}$$

In the derivation of inequality (3.17), we assume that after robot i moves from s_i to $s_i^\star$, the resultant strategy profile still satisfies the constraint (e.g, the number of a task's participating robots does not exceed the constraint on the maximum number of participating robots). However, it is possible that the resultant strategy profile does not satisfies the constraints. As shown in equation (3.4), a robot's payoff would become zero if the strategy profile does not satisfies the constraints. To handle this case, we let λ and μ satisfy additional constraints for arbitrary task $j \in T$:

$$l_j(n_j) \geq \lambda l_j(n_j^\star) n_j^\star - \mu l_j(n_j) n_j, \quad n_j = \{1, \ldots, c_j\}, \quad n_j^\star = \{1, \ldots, c_j\} \tag{3.22}$$

where c_j is the constraint on the maximum number of participating robots of task j.

Suppose there is a set $N^{'}$ of robots switching to task j in the strategy profile $\boldsymbol{s}^{'}$, but the resultant strategy profile would violate the constraint. The reason why (3.22) works is as

follows:

$$\begin{aligned}\sum_{i\in N'} P_i(\boldsymbol{s}) &= \sum_{i\in N'} l_{\boldsymbol{s}_i}(n_{\boldsymbol{s}_i}) \\ &\geq \lambda l_j(n_j^\star)n_j^\star - \mu l_j(n_j)n_j\end{aligned} \tag{3.23}$$

where $\boldsymbol{s}_i$ is the task robot i chooses in the strategy profile $\boldsymbol{s}$ and $n_{\boldsymbol{s}_i}$ is the number of robots doing task $\boldsymbol{s}_i$ in the strategy profile $\boldsymbol{s}$. The inequality holds true due to (3.22). We can plug (3.23) into (3.20) such that (3.21) still holds true.

In summary, for this specific resource allocation problem, one way to find the lower bound of suboptimality is to solve the following nonlinear optimization problem with linear constraints:

$$\begin{aligned}&\max_{\lambda>0,\mu>0} \quad \frac{\lambda}{1+\mu} \\ &s.t. \qquad \lambda q^2 - \mu p^2 \leq pq, \quad p = \{1,\ldots,(c_k)_{max}-1\}, q = \{1,\ldots,(c_k)_{max}\} \\ &\qquad \lambda (q')^2 - \mu (p')^2 \leq p', \quad p' = \{1,\ldots,(c_k)_{max}\}, q' = \{1,\ldots,(c_k)_{max}\}\end{aligned} .$$

where $(c_k)_{max}$ is the maximum of $c_k, \forall k \in T$ and c_j is the constraint on the maximum number of participating robots of task j.

Example 1. *Suppose there are twenty robots and seven tasks and that the maximum number for each task's participating robot is 3, i.e.* $c_i = 3$, $i = \{1, ..., 7\}$. *The corresponding suboptimality lower bound is* 0.1111.

Remark 3. *Suboptimality lower bounds obtained by solving the nonlinear optimization problem are conservative. In our experiments, we observe much better performances, which are close to optimal.*

3.6.2 Marginal Gain Decreasing

For this specific resource allocation problem formulated in section 3.2.1, robots' payoff functions are monotonically decreasing with respect to the number of co-working robots

(see section 3.2.1). The suboptimality lower bound is at least $\frac{1}{2}$ for any pure Nash equilibria [8].

3.7 Simulations

3.7.1 Scalability

We conduct numerical simulations to investigate algorithms' scalability regarding the number of tasks and robots. We define an iteration as an iteration of the main loop of the algorithms (lines 10-24 for sequential best response dynamics, lines 9-16 for concurrent best response dynamics, lines 9-30 for decentralized dual greedy algorithm). For each test case, we run 100 Monte Carlo trials. The statistical results of iterations required for convergences are shown in figures 3.4-3.7 as box plots. The mean values of iterations are indicated by a green circle and connected via green lines.

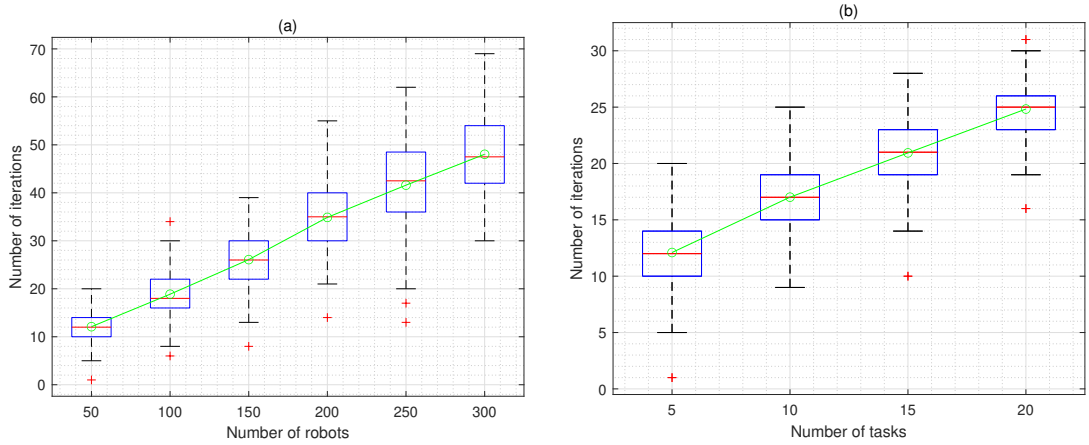


Figure 3.4: Convergence performance for sequential best response dynamics with decreasing marginal gain. (a)iterations of convergence for varying numbers of tasks; (b)iterations of convergence for varying numbers of robots

We test the scalability of sequential best response dynamics applying to both two types of resource allocation problems. For resource allocation problems with decreasing marginal gain, the convergence performance is shown in figure 3.4. To test the scalability regarding the number of robots, we consider scenarios with fixed $t = 5$ and $n \in$

$\{50, 100, 150, 200, 250, 300\}$. Figure 3.4(a) shows the number of iterations for convergence is approximately a linear function of the number of robots, which lines up with the analysis in section 3.3.3. For scalability with respect to the number of tasks, we consider scenarios with fixed $n = 50$ and $t \in \{5, 10, 15, 20\}$. Figure 3.4(b) shows that the number of iterations for convergence is also approximately a linear function of the number of tasks. However, the analysis in section 3.3.3 states that iterations are quadratic in the number of tasks. The reason is that analysis in section 3.3.3, which considers worst cases, is conservative. For resource allocation problems with increasing marginal gain, the convergence performance is shown in figure 3.5. To test the scalability regarding the number of robots, we consider scenarios with fixed $t = 5$ and $n \in \{50, 100, 150, 200, 250, 300\}$. Figure 3.5(a) shows the number of iterations for convergence is approximately a linear function of the number of robots. For scalability with respect to the number of tasks, we consider scenarios with fixed $n = 300$ and $t \in \{5, 10, 15, 20\}$. Figure 3.5(b) shows that the number of iterations is linear with the number of tasks.

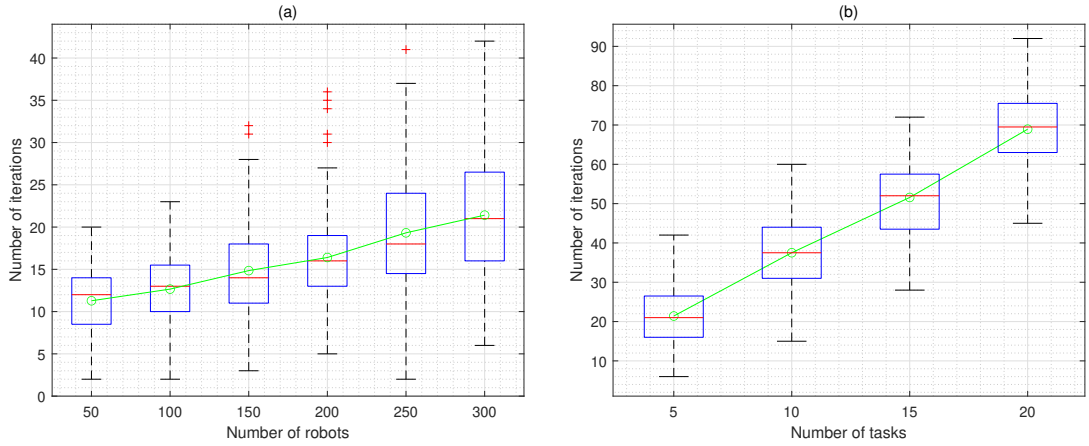


Figure 3.5: Convergence performance for sequential best response dynamics with increasing marginal gain and constraints on maximum participating robots: (a)iterations of convergence for varying numbers of tasks; (b)iterations of convergence for varying numbers of robots increase min

Test results for decentralized dual greedy algorithm are shown in figure 3.6. Given the number of robots and tasks, we observe that the iterations required for convergence are

fixed in the experiments. Test results clearly indicate the number of iterations for convergence is both linear in the number of tasks and robots. This aligns with the analysis in section 3.5.3.

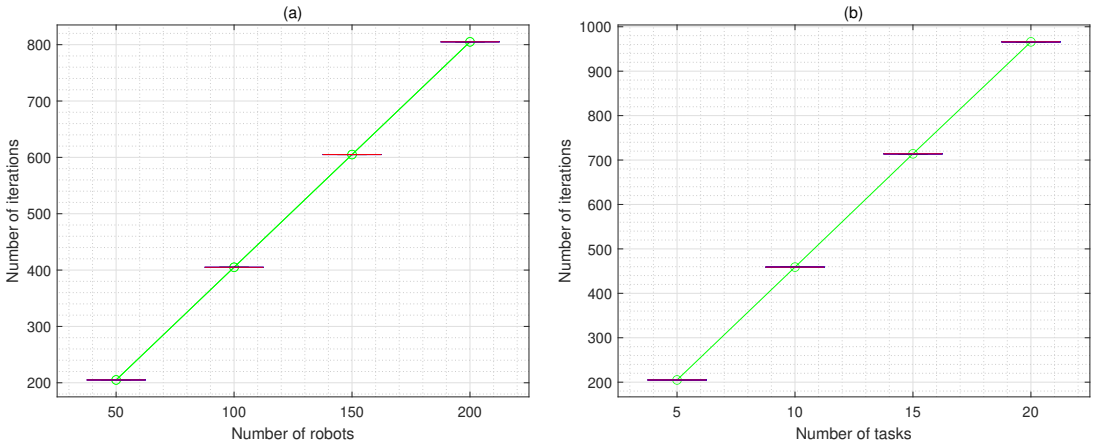


Figure 3.6: Convergence performance for decentralized dual greedy with decreasing marginal gain. (a)iterations of convergence for varying numbers of tasks; (b)iterations of convergence for varying numbers of robots

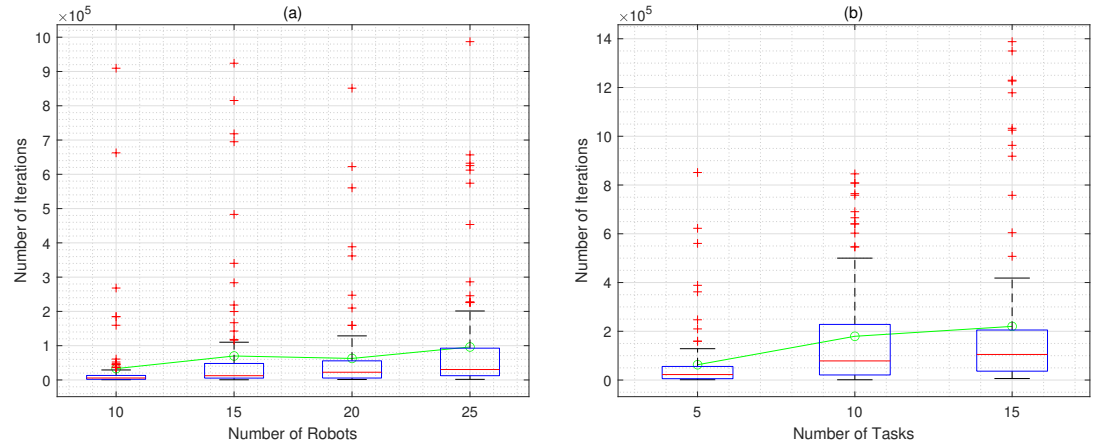


Figure 3.7: Convergence performance for concurrent best response dynamics with decreasing marginal gain. (a)iterations of convergence for varying numbers of tasks; (b)iterations of convergence for varying numbers of robots

Test results for concurrent best response dynamics are shown in figure 3.7. For scalability with respect to the number of robots, we consider scenarios with fixed $t = 5$ and $n \in \{10, 15, 20, 25\}$. For scalability regarding the number of tasks, we consider scenarios with fixed $n = 20$ and $t \in \{5, 10, 15\}$.

3.7.2 Suboptimality

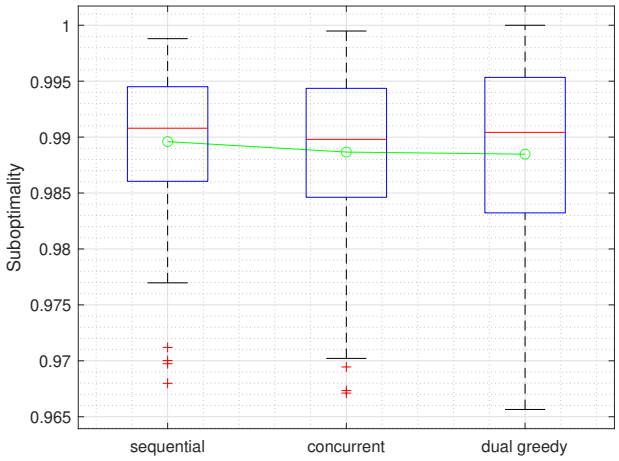


Figure 3.8: Suboptimality trials for resource allocation problems with decreasing marginal gains

We apply sequential best response dynamics, concurrent best response dynamics, and decentralized dual greedy algorithm to resource allocation problems with decreasing marginal gains (section 3.2.1). We consider scenarios with 50 robots and 5 tasks. Task weights are randomly generated. Figure 3.8 shows all algorithms achieve near-optimal performances.

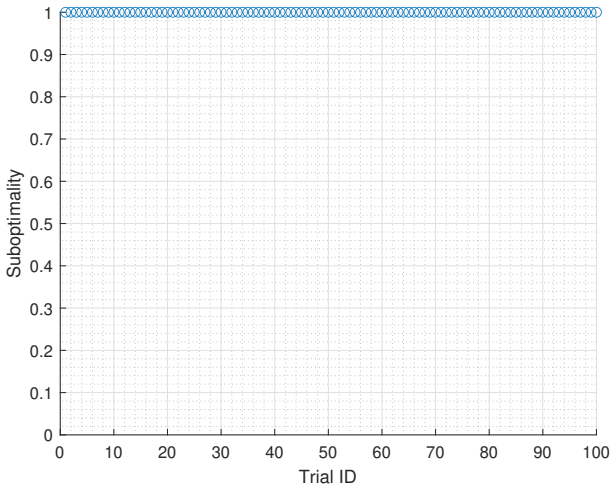


Figure 3.9: Suboptmality lower bounds for resource allocation problems with increasing marginal gains

For resource allocation problems with increasing marginal gains, we apply decentralized sequential best response dynamics to give approximate solutions. We consider 20 robots and 7 tasks. Each task could have a maximum of three participating robots. We run 100 Monte Carlo trials (see figure 3.9). The suboptimality lower bound computed by solving the nonlinear optimization problem discussed in section 3.6.2 is 0.1111. However, in the experiment, we observe optimal performances. This validates the statement in remark 4- the suboptimality lower bound is conservative.

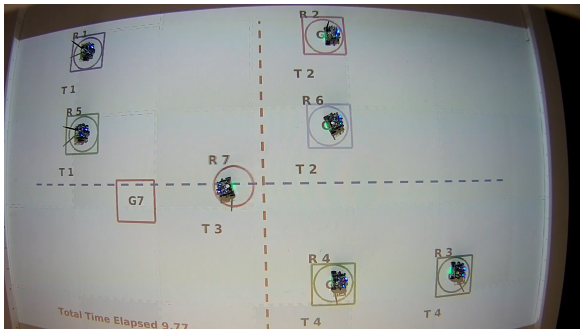
3.7.3 Comparison

Decentralized sequential best response dynamics could admit all classes of payoff functions, while concurrent best response dynamics and dual greedy could only admit payoff functions that are monotonically decreasing with respect to the number of participating robots. Concurrent best response dynamics require less coordination than sequential best response dynamics. However, concurrent best response dynamics needs much more iterations to converge compared to sequential best response dynamics. Three algorithms have not many differences in terms of suboptimality.

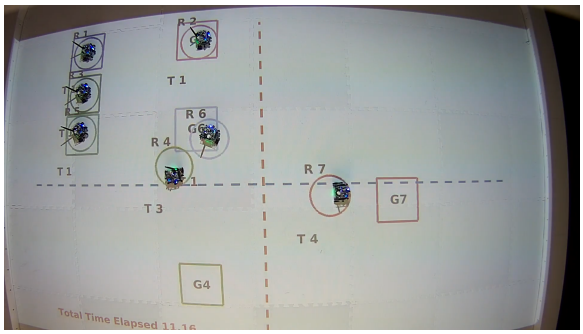
3.8 Experiments on Robotarium

In this section, we implement sequential best response, concurrent best response, and decentralized dual greedy algorithm on the multi-robot test bed Robotarium.

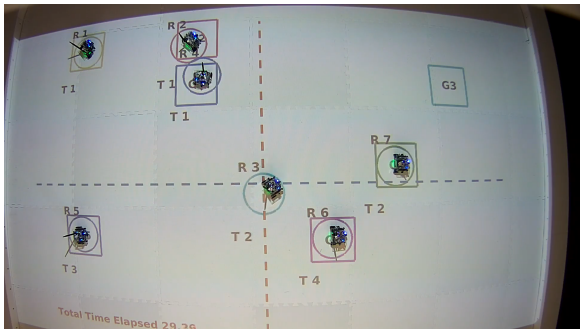
We consider 7 robots and 4 tasks. We divide the Robotarium into 4 subareas. Each subarea represents a task. If a robot is staying in a subarea, it is participating in the task which this specific subarea represents. Figure 3.10 shows decision processes for three different tasks. For resource allocation problems with decreasing marginal gains, we implement sequential best response dynamics, concurrent best response dynamics and decentralized dual greedy to give approximate solutions. The task weights for 4 tasks are $[0.8, 0.7, 0.6, 0.9]$. All three algorithms converge to the same pure Nash equilibrium. See table 3.1 for task



(a)



(b)



(c)

Figure 3.10: (a) Decision process for sequential best response dynamics; (b) Decision process for concurrent best response dynamics (c) Decision process for decentralized dual greedy algorithm

allocation result.

Table 3.1: Task allocation result for resource allocation problems with decreasing marginal gains

	Task 1	Task 2	Task 3	Task 4
Sequential best response dynamics	2	1	1	3
Concurrent best response dynamics	2	1	1	3
Decentralized dual greedy	2	1	1	3

For resource allocation with increasing marginal gains, we implement sequential best response dynamics to give approximate solutions. Weights and maximum number of participating robots are $[0.8, 0.7, 0.6, 0.9]$ and $[2, 2, 2, 2]$. See table 3.2 for task allocation result.

Table 3.2: Task allocation result for resource allocation problems with increasing marginal gains

	Task 1	Task 2	Task 3	Task 4
Sequential best response dynamics	2	2	1	2

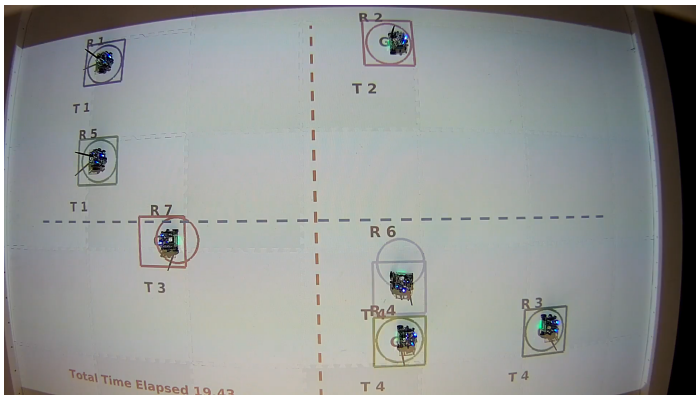


Figure 3.11: Pure Nash equilibrium for resource allocation problems with decreasing marginal gains

CHAPTER 4
WEIGHTED CONGESTION GAME-BASED TASK ALLOCATION FOR HETEROGENEOUS ROBOT TEAMS

In this chapter, we illustrate how to utilize weighted congestion game theory to solve task allocation problems for heterogeneous robot teams. We model the system of heterogeneous robots as a community of different species (different types) of robots. Thanks to weighted congestion game theory, different species of robots have different weights for different tasks to capture their different impacts on different tasks. We model the task allocation problem as two different types of resource allocation problems. We prove the decentralized sequential best response dynamic would converge to a pure Nash Equilibrium under the framework of weighted congestion game theory. Lower bounds of suboptimality of pure Nash equilibria with respect to the resource allocation problems are discussed. Numerical simulations and experiments on Robotarium are conducted to show the effectiveness of the above algorithms. We also show that strong Nash equilibrium is not equal to pure Nash equilibriumm in the settings of ST-MR task allocation problems and the potential of strong Nash equilibrium to improve suboptimality lower bound.

4.1 Preliminaries

Definition 8. *A weighted congestion game model is a tuple* $(N, T, E, \widetilde{N}, (\Sigma_i)_{i\in N}, (w_i)_{i\in N}, W, (\overline{l_j})_{j\in T}, (P_i)_{i\in N})$*, where (1)* $N = \{1, 2, ..., n\}$ *denotes the set of* n *robots; (2)* $T = \{1, 2, ..., t\}$ *denotes the set of* t *different tasks; (3)* $E = \{1, 2, ..., e\}$ *denotes the set of* e *different species of robots; (4)* $\widetilde{N} = \{\widetilde{n}_1, ..., \widetilde{n}_e\}$*, where* $\widetilde{n}_1, ..., \widetilde{n}_e$ *are the number of robots for species* $1, ..., e$*; (5)* $\Sigma_i \subseteq T$ *is the strategy space for robot* i*, which contains all tasks robot* i *could choose; (6)* $w_i = \{w_{i1}, ..., w_{it}\}$ *is the weight profile for robot* i*, where* $w_{ij} \in [0, 1]$ *is robot* i*'s weight for task* $j \in T$ *(see figure 4.1 for an illustration of*

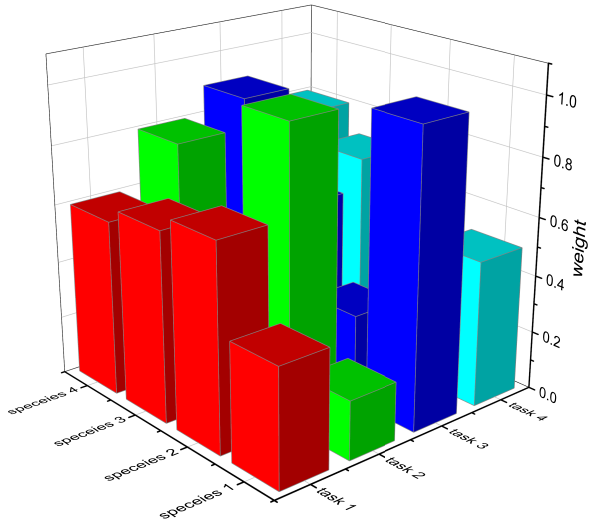


Figure 4.1: an example of weight profiles for 4 species of robots and 4 different tasks

weight profile). Robots belong to the same species have the same weight profile; (7) $W = \{W_1, ..., W_t\}$*, where* $W_1, ..., W_t$ *are the total weights of robots (sum of weights of robots doing this specific task) assigned to task* $1, ..., t$*; (8)* $\overline{l_j}$ *is the latency function for task* j*, and the latency of a task measures the congestion of that task, which depends on how much weights are assigned to that task; (9)* P_i *is the payoff function for robot* i*.*

Denote $\boldsymbol{S} = \Sigma_1 \times \Sigma_2 ... \times \Sigma_N$*. A strategy profile of the standard congestion game is a vector* $\boldsymbol{s} = (s_1, ..., s_n) \in \boldsymbol{S}$*, where robot* $i \in N$ *chooses a task* $s_i \in \Sigma_i$*. The latency function* $\overline{l_j(\boldsymbol{s})}$ *for task j is a function mapping from* $\boldsymbol{S}$ *to* $\mathbb{R}$*. Given a strategy profile* s*,* W_j *is the total weights of robots assigned to task* j*. The latency function* $\overline{l_j(\boldsymbol{s})}$ *for task j only depends on* W_j *such that* $\overline{l_j(\boldsymbol{s})} = l_j(W_j)$*.*

Given a strategy profile $\boldsymbol{s}$*, Robot i's payoff function* $P_i(\boldsymbol{s})$ *is equal to task* s_i*'s latency function such that* $P_i(\boldsymbol{s}) = \overline{l_{s_i}(\boldsymbol{s})} = l_{s_i}(W_{s_i})$*, where* s_i *is the task robot i is assigned to in strategy profile* $\boldsymbol{s}$*. Let* $J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s})$*. For* $J(\boldsymbol{s})$*, we have the following relationship:*

$$J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) = \sum_{j \in T} n_j \overline{l_j(\boldsymbol{s})} = \sum_{j \in T} n_j l_j(W_j)$$

We consider payoff maximization games, in which robots strive to maximize their own payoff.

Definition 9. *A weighted potential function is a function Φ mapping from $\boldsymbol{S}$ to $\mathbb{R}$, such that for $\forall i \in N$ and $\forall \boldsymbol{s} = (s_1, ..., s_i, ..., s_n), \boldsymbol{s}' = (s_1, ..., s_i', s_{i+1}, ..., s_n) \in \boldsymbol{S}$,*

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') = k(P_i(\boldsymbol{s}) - P_i(\boldsymbol{s}')).$$

where $k \in \mathbb{R}_{>0}$ is a constant. Put it in another way, if a robot unilaterally changes its task, the change of the exact potential function is equal to the change of the robot's payoff.

Definition 10. *A congestion game is a weighted potential game if there exists a weighted potential function.*

4.2 Two Resource Allocation Problems for Task Allocation for Heterogeneous Robot Teams

As discussed in section 3.2, we formulate the task allocation problem as two types of resource allocation problems. The first resource allocation problem assumes the better performance of the task with more participating robots. However, the corresponding marginal gain with respect to the number of participating robots is monotonically decreasing. The second resource allocation problem assumes the better performance of the task with more participating robots, and the corresponding marginal gain regarding the number of participating robots is monotonically increasing.

4.2.1 Marginal Gain Increasing With Respect to the Number of Participating Robots

For this type of resource allocation problem, the corresponding optimization problem is formulated:

$$\begin{aligned} \max_{W_1,...,W_t} \quad & \sum_{j=1}^{t} a_j W_j^2 \\ \text{s.t.} \quad & n_j \leq c_j, \quad j = \{1, 2, ..., t\} \end{aligned} \tag{4.1}$$

where $W_1, ..., W_t$ is the total weights of robots (sum of weights of robots doing this specific task) assigned to task $1, ..., t$, n_j is the number of robots participating in task j, $\sum_{j=1}^{t} n_j = n$, $a_j \in [0, 1]$ is the associated weight (priority) of task j, c_j is the constraint on the maximum number of participating robots for task j, and $\sum_{j=1}^{t} c_j \geq n$.

This optimization problem is known as multiple quadratic knapsack problem, which is NP-hard [38].

Given a strategy profile $\boldsymbol{s}$, the corresponding robot level payoff function for robot i is:

$$P_i(\boldsymbol{s}) = \overline{l_{s_i}(\boldsymbol{s})} = l_{s_i}(W_{s_i}) = \begin{cases} a_{s_i} w_{is_i} W_{s_i} & \text{if } n_{s_i} \leq c_{s_i} \\ 0 & \text{if } n_{s_i} > c_{s_i} \end{cases} \tag{4.2}$$

where s_i is the task robot i chooses in the strategy profile $\boldsymbol{s}$, l_{s_i} is the latency function of task s_i, w_{is_i} is robot i's weight for task s_i, W_{s_i} is the total weights of robots doing task s_i, n_{s_i} is the number of robots participating in task s_i. The robot level payoff function $P_i(\boldsymbol{s})$ is monotonically increasing with respect to the total weights of co-working robots when the number of co-working robots is no more than c_{s_i}. If the number of co-working robots is greater than c_{s_i}, robot i's payoff becomes zero. This ensures the numbers of participating robots for all tasks satisfy the constraints in any resultant pure Nash equilibria. By summing over all robots' payoff functions, optimizing the sum of robots' payoff become equivalent to optimizing the above resource allocation problem. To be more specific, we have the

following relationship:

$$J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) = \sum_{j \in T} n_j l_j(W_j) = \sum_{j=1}^{t} a_j W_j^2$$

.

4.2.2 Marginal Gain Decreasing With Respect to the Number of Participating Robots

For this resource allocation problem, the optimization problem is formulated as:

$$\max_{W_1, W_2, ..., W_t} \quad \sum_{j=1}^{t} a_j \frac{n_j \log(1 + W_j)}{W_j} \tag{4.3}$$

where $W_1, ..., W_t$ is the total weights of robots (sum of weights of robots doing this specific task) assigned to task $1, ..., t$, n_j is the number of robots participating in task j, $\sum_{j=1}^{t} n_j = n$ and $a_j \in [0, 1]$ is the associated weight (priority) of task j.

The ideal objective function is $\sum_{j=1}^{t} a_j \log(1 + W_j)$. However, there are no theoretical guarantees of convergences to pure Nash equilibriums for this objective function. Although jiggling at some points, the objective function in (4.3) could be viewed as an approximation to the ideal objective function, which has theoretical guarantees of convergences to pure Nash equilibriums (see figure 4.2).

Given a strategy profile $\boldsymbol{s}$, the corresponding robot level payoff function for robot i is:

$$P_i(\boldsymbol{s}) = \overline{l_{\boldsymbol{s}_i}(\boldsymbol{s})} = l_{\boldsymbol{s}_i}(W_{s_i}) = \begin{cases} \frac{a_{\boldsymbol{s}_i} \log(1 + W_{\boldsymbol{s}_i})}{W_{\boldsymbol{s}_i}} & \text{if } W_{\boldsymbol{s}_i} > 0 \\ 0 & \text{if } W_{\boldsymbol{s}_i} = 0 \end{cases} \tag{4.4}$$

where s_i is the task robot i chooses in the strategy profile $\boldsymbol{s}$, l_{s_i} is the latency function of task k and W_{s_i} is the total weights of robots doing task s_i. The robot level payoff function $P_i(\boldsymbol{s})$ is monotonically decreasing with respect to the total weights of co-working robots. By summing over all robots' payoff functions, optimizing the sum of robots' payoff become

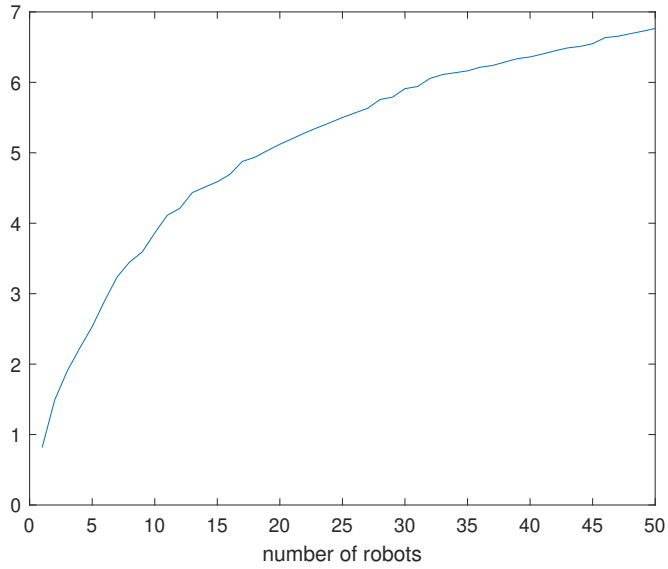


Figure 4.2: an example of the social payoff of a task in (4.3), i.e, the sum of all participating robots' payoff

equivalent to optimizing the above resource allocation problem. To be more specific, we have the following relationship:

$$J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) = \sum_{j \in T} n_j l_j(W_j) = \sum_{j=1}^{t} a_j \frac{n_j \log(1 + W_j)}{W_j}$$

.

4.3 Decentralized sequential best response dynamics

4.3.1 Algorithm Description

Readers are referred to section 3.3.1 for pseudo code and algorithm description.

4.3.2 Convergence to a Pure Nash Equilibrium

Definition 11. *Sorted lexicographical order: Let* $\boldsymbol{a} = (a_1, ..., a_q)$, $\boldsymbol{b} = (b_1, ..., b_q)$ *be two vectors* $\in \mathbb{R}^q_+$. *Let* $\hat{\boldsymbol{a}} = (\hat{a_1}, ..., \hat{a_q})$ *and* $\hat{\boldsymbol{b}} = (\hat{b_1}, ..., \hat{b_q}) \in \mathbb{R}^q_+$ *be two lexicographically sorted vectors generated from* $\boldsymbol{a}$, $\boldsymbol{b}$ *by permuting the entries in a non-decreasing way such*

that $\hat{a_1} \geq \hat{a_2} \geq ... \geq \hat{a_q}$ *and* $\hat{b_1} \geq \hat{b_2} \geq ... \geq \hat{b_q}$. $\boldsymbol{a}$ *is sorted lexicographically smaller than* $\boldsymbol{b}$ *if there exists an index m such that* $\hat{a_i} = \hat{b_i}$ *for all* $i < m$ *and* $\hat{a_m} < \hat{b_m}$ *[47].*

Theorem 5. *Decentralized sequential best response dynamics converge to a pure Nash equilibrium for the two resource allocation problems formulated in section 4.2.*

Proof. (a) Marginal gain increasing with respect to the number of participating robots

For this type of resource allocation problem, we show best response dynamics converge to a pure Nash equilibrium by constructing a weighted potential function.

Consider a weighted potential function:

$$\Phi(\boldsymbol{s}) = \sum_{j=1}^{t} a_j W_j^2 + \sum_{i=1}^{n} a_k w_{ik}^2 \tag{4.5}$$

where W_j is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}$, k is the task robot i is participating in strategy profile $\boldsymbol{s}$, w_{ik} is robot i's weight for task k and a_j, a_k are the associated weights (priority) for task j and k.

To show $\Phi(\boldsymbol{s})$ is a weighted potential function, consider a robot i moves from task p to task q. The strategy profile before robot i's move is $\boldsymbol{s} = (s_1, ..., s_i = p, ..., s_n)$ and the strategy profile after robot i's move is $\boldsymbol{s}' = (s_1, ..., s_i = q, ..., s_n)$. We have:

$$\begin{aligned} \Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') &= a_p W_p^2 - a_p (W_p - w_{ip})^2 + a_q W_q^2 - a_q (W_q + w_{iq})^2 + a_p w_{ip}^2 - a_q w_{iq}^2 \\ &= 2(a_p w_{ip} W_p - a_q w_{iq} (W_q + w_{iq})) \end{aligned} \tag{4.6}$$

where $(a_p w_{ip} W_p - a_q w_{iq} (W_q + w_{iq})$ is equal to $P_i(\boldsymbol{s}) - P_i(\boldsymbol{s}')$. Hence, $\Phi(\boldsymbol{s})$ is a weighted potential function.

Note that $\Phi(\boldsymbol{s})$ is a potential function for strategy profiles that satisfy the constraints (i.e., constraints on the maximum number of participating robots for each task) discussed in section 4.2.1. Although the strategy profile generated after the initialization phase may not satisfy the constraints, robots' special tailored payoff functions (see equation (4.2)) would lead to strategy profiles that satisfy the constraints. Hence, we can safely ignore

the possibility that strategy profiles generated after the initialization phase may not satisfy the constraints, and sequential best response dynamics would converge to a pure Nash equilibrium.

(b) Marginal gain decreasing with respect to the number of participating robots

For this type of resource allocation problem, we show that sequential best response dynamics would converge to a pure Nash equilibrium by showing it is monotonically decreasing in the sorted lexicographical order.

Given a strategy profile $\boldsymbol{s} = (s_1, ..., s_i = p, ..., s_n)$, vector $L(\boldsymbol{s}) = (W_1, ..., W_t)$, where $W_1, ..., W_t$ are the sum of weights of task $1, ..., t$'s participating robots in strategy profile $\boldsymbol{s}$. Suppose a robot i moves from task p to task q under sequential best response dynamics. The strategy profile after robot i's move is $\boldsymbol{s}^{'} = (s_1, ..., s_i = q, ..., s_n)$ while $L(\boldsymbol{s}^{'}) = (W_1, ..., W_p - w_{ip}, ..., W_q + w_{iq}, ..., W_t)$. Let $\hat{L(\boldsymbol{s})}$ and $\hat{L(\boldsymbol{s}')}$ be the lexicographically sorted vectors generated form $L(\boldsymbol{s})$ and $L(\boldsymbol{s}^{'})$. Since robot i moves from task p to task q under best response dynamics, $\frac{\log(1+W_p)}{W_p}$ is less than $\frac{\log(1+W_q+w_{iq})}{W_q+w_{iq}}$. Hence, $W_p > W_q + w_{iq}$. Also W_p is bigger than $W_p - w_{ip}$. As a result, $\hat{L(\boldsymbol{s}')}$ is sorted lexicographically smaller than $\hat{L(\boldsymbol{s})}$. Therefore, strategy profiles generated under sequential best response dynamics are monotonically decreasing in the sorted lexicographic order. In conclusion, sequential best response dynamics converge to a pure Nash equilibrium.

□

4.3.3 Time Complexity Analysis

We refer to the unit of time required for each robot to execute an iteration of the sequential best response dynamics (lines 10-24) as a time step. Recall in definition 8, $\widetilde{N} = \{\widetilde{n}_1, ..., \widetilde{n}_e\}$, where $\widetilde{n}_1, ..., \widetilde{n}_e$ are the number of robots for species $1, ..., e$. Also $W = \{W_1, ..., W_t\}$, where $W_1, ..., W_t$ are the total weights of robots (sum of weights of robots doing this specific task) assigned to task $1, ..., t$. Given $\widetilde{N}$, there are $\prod_{m=1}^{e} \binom{\widetilde{n}_e+t-1}{t-1}$ different configurations of W. For the case of marginal gain decreasing with respect to the number

of participating robots, the maximum number of time steps required for converging to a pure Nash equilibrium is $\prod_{m=1}^{e} \binom{\widetilde{n}_e+t-1}{t-1}$. For the case of marginal gain increasing with respect to the number of participating robots, it might take $\prod_{m=1}^{e} \binom{\widetilde{n}_e+t-1}{t-1}$ time steps to reach a configuration which satisfies the constraints. Next, it would take at most $\prod_{m=1}^{e} \binom{\widetilde{n}_e+t-1}{t-1}$ iterations to reach a pure Nash equilibrium. In a single time step, a robot needs to investigate t tasks in an iteration (line 11). The complexity for constructing the set $K^\star$ and finding $k^{\star\star}$ is O(n) (lines 20-21). In conclusion, the time complexity is $O(nt \prod_{m=1}^{e} \binom{\widetilde{n}_e+t-1}{t-1})$.

4.4 Suboptimality

Recall the definition of suboptimality in definition 7. In this section, we investigate the lower bound of suboptimality.

4.4.1 Marginal Gain Decreasing

We utilize the $\lambda - \mu$ smoothness technique developed in [10] to investigate lower bound of suboptimality.

Assumption 5. *For a pure Nash equilibrium strategy profile $\boldsymbol{s}$, $W_j \geq 1, \forall j \in T$. In other words, each task's total weight is no less than one.*

Assumption 6. *For the optimal strategy strategy profile $\boldsymbol{s}^\star$, $W_j^\star \geq 1, \forall j \in T$, where $W_j^\star$ is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}^\star$. In other words, each task's total weight is no less than one.*

These two assumptions are useful for the further derivation.

Given a strategy profile $\boldsymbol{s}$ which is a pure Nash equilibrium and a strategy profile $\boldsymbol{s}^\star$ which maximizes the sum of all robots' payoff. By the definition of pure Nash equilibrium, we have the following inequality for any robot i:

$$P_i(\boldsymbol{s}) \geq P_i(\boldsymbol{s}^{'}) \tag{4.7}$$

where $\boldsymbol{s} = (s_1, ..., s_i, ..., s_n)$, $\boldsymbol{s}^{'} = (s_1, ..., s_i^{\star}, ..., s_n)$ and $s_i^{\star}$ is the task robot i chooses in the optimal strategy profile $\boldsymbol{s}^{\star}$.

Thanks to the hierarchical structure of congestion games, we could sum all robots' payoff:

$$\begin{aligned} J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) &\geq \sum_{i \in N} P_i(\boldsymbol{s}^{'}) \\ &\geq \sum_{\{j \in T | n_j^{\star} > 0\}} l_j(W_j + 1) n_j^{\star} \\ &\geq \sum_{\{j \in T | n_j^{\star} > 0\}} l_j(\widetilde{W_j} + 1) n_j^{\star} \end{aligned} \tag{4.8}$$

where $J_{\boldsymbol{s}} = \sum_{i \in N} P_i(\boldsymbol{s})$, W_j is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}$, $n_j^{\star}$ is the number of robots doing task j in strategy profile $\boldsymbol{s}^{\star}$, and $\widetilde{W_j}$ is the integer generated by applying the ceiling operator to W_j. The first inequality holds true due to (4.7). The second and third inequality holds true as l_j, $\forall j \in T$, is a monotonically decreasing function for this specific resource allocation problem.

Suppose there exists $\lambda > 0$ and $\mu > 0$ such that for arbitrary task j:

$$\begin{aligned} &l_j(\widetilde{W_j} + 1) n_j^{\star} \geq \lambda l_j(\underset{\sim\sim}{W_j^{\star}}) n_j^{\star} - \mu l_j(\widetilde{W_j}) n_j, \quad n_j = \{1, \ldots, n\}, \quad n_j^{\star} = \{1, \ldots, n\}, \\ &\widetilde{W_j} = \{1, ..., n_j\}, \quad \underset{\sim\sim}{W_j^{\star}} = \{1, ..., max\{1, n_j^{\star} - 1\}\} \end{aligned} \tag{4.9}$$

where n_j is the number of robots doing task j in strategy profile $\boldsymbol{s}$, $W_j^{\star}$ is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}^{\star}$,$\underset{\sim\sim}{W_j^{\star}}$ is the integer generated by applying the floor operator to $W_j^{\star}$. W_j and $W_j^{\star}$ are both ranging from 1 to n instead of ranging from 0 to n due to assumption 5 and 6.

Then we have:

$$
\begin{aligned}
J(\boldsymbol{s}) &\geq \sum_{\{j\in T|n_j^\star>0\}} l_j(\widetilde{W_j}+1)n_j^\star \\
&\geq \sum_{\{j\in T|n_j^\star>0\}} \lambda l_j(\underset{\sim\sim}{W_j^\star})n_j^\star - \mu l_j(\widetilde{W_j})n_j \\
&\geq \sum_{\{j\in T|n_j^\star>0\}} \lambda l_j(W_j^\star)n_j^\star - \mu l_j(W_j)n_j \\
&\geq \sum_{\{j\in T\}} \lambda l_j(W_j^\star)n_j^\star - \mu l_j(W_j)n_j \\
&= \lambda J(\boldsymbol{s}^\star) - \mu J(\boldsymbol{s})
\end{aligned}
\tag{4.10}
$$

where $J(\boldsymbol{s}) = \sum_{i\in N} P_i(\boldsymbol{s})$, $J(\boldsymbol{s}^\star) = \sum_{i\in N} P_i(\boldsymbol{s}^\star)$, W_j is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}$, $W_j^\star$ is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}^\star$, $n_j^\star$ is the number of robots doing task j in strategy profile $\boldsymbol{s}^\star$, n_j is the number of robots doing task j in strategy profile $\boldsymbol{s}$, $\widetilde{W_j}$ is the integer generated by applying the ceiling operator to W_j and $\underset{\sim\sim}{W_j^\star}$ is the integer generated by applying the floor operator to $W_j^\star$. The first inequality holds true due to inequality (4.8). Then second inequality is true as a result of inequality (4.9). The third inequality holds true as l_j, $\forall j \in T$, is a monotonically decreasing function for this specific resource allocation problem. The forth inequality holds ture as $\mu > 0$.

Hence, we get a lower bound of suboptimality for any given pure Nash equilibrium $\boldsymbol{s}$:

$$
\alpha = \frac{J(\boldsymbol{s})}{J(\boldsymbol{s}^\star)} \geq \frac{\lambda}{1+\mu} \tag{4.11}
$$

In summary, for this specific resource allocation problem, one way to find the lower bound of suboptimality is to solve the following nonlinear optimization problem with linear

constraints:

$$\max_{\lambda>0,\mu>0} \quad \frac{\lambda}{1+\mu}$$
$$s.t. \qquad \lambda\frac{\log(1+W_q)}{W_q}q - \mu\frac{\log(1+W_p)}{W_p}p \leq \frac{\log(1+W_p)}{W_p}q, \quad p=\{1,...,n\},$$
$$q=\{1,\ldots,n\}, W_p=\{1,...,p\}, W_q=\{1,...,max\{1,q-1\}\}.$$

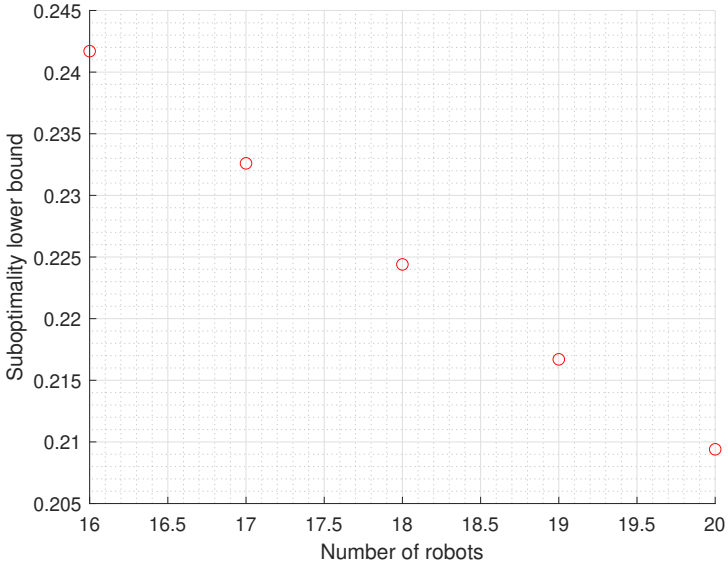


Figure 4.3: Suboptimality lower bounds obtained for n=16,17,18,19,20 robots

Example 2. *Figure 4.3 shows suboptimality lower bounds for n=16,17,18,19 and 20 robots by solving the nonlinear optmization problem.*

4.4.2 Marginal Gain Increasing

Assumption 7. *For a pure Nash equilibrium strategy profile* $\boldsymbol{s}$*,* $W_j \geq 1, \forall j \in T$*. In other words, each task's total weight is no less than one.*

Assumption 8. *For the optimal strategy strategy profile* $\boldsymbol{s}^\star$*,* $W_j^\star \geq 1, \forall j \in T$*. In other words, each task's total weight is no less than one.*

These two assumptions are useful for further derivation.

Suppose strategy profile $\boldsymbol{s}$ is a pure Nash equilibrium, and $\boldsymbol{s}^\star$ is a strategy profile that maximizes the sum of all robots' payoff. By the definition of pure Nash equilibrium, we have the following inequality for any robot i:

$$P_i(\boldsymbol{s}) \geq P_i(\boldsymbol{s}^{'}) \tag{4.12}$$

where $\boldsymbol{s} = (s_1, ..., s_i, ..., s_n)$, $\boldsymbol{s}^{'} = (s_1, ..., s_i^\star, ..., s_n)$ and $s_i^\star$ is the task robot i chooses in the optimal strategy profile $\boldsymbol{s}^\star$. For now, we assume after robot i moves from s_i to $s_i^\star$, the resultant strategy profile still satisfies the constraint (e.g, the number of a task's participating robots does not exceed the constraint on the maximum number of participating robots).

Thanks to the hierarchical structure of congestion games, we could sum all robots' payoff:

$$\begin{aligned} J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s}) &\geq \sum_{i \in N} P_i(\boldsymbol{s}^{'}) \\ &\geq \sum_{\{j \in T | n_j^\star > 0\}} l_j(W_j) n_j^\star \\ &\geq \sum_{\{j \in T | n_j^\star > 0\}} l_j(\underbrace{W_j}) n_j^\star \end{aligned} \tag{4.13}$$

where $J(\boldsymbol{s}) = \sum_{i \in N} P_i(\boldsymbol{s})$, W_j is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}$, $n_j^\star$ is the number of robots doing task j in strategy profile $\boldsymbol{s}^\star$, and $\underbrace{W_j}$ is the integer generated by applying the floor operator to W_j. The first inequality holds due to (4.12). The second and third inequality hold as a result of the fact that for $\forall j \in T$, l_j is a monotonically increasing function for this specific resource allocation problem.

Suppose there exists $\lambda > 0$ and $\mu > 0$ such that for arbitrary task j:

$$\begin{aligned} &l_j(\underbrace{W_j}) n_j^\star \geq \lambda l_j(\widetilde{W_j^\star}) n_j^\star - \mu l_j(\underbrace{W_j}) n_j, \quad n_j = \{1, \ldots, c_j - 1\}, \\ &n_j^\star = \{1, \ldots, c_j\}, \quad \underbrace{W_j} = \{1, ..., max\{n_j - 1, 1\}\}, \quad \widetilde{W_j^\star} = \{1, ..., n_j^\star\} \end{aligned} \tag{4.14}$$

where c_j is the constraint on the maximum participating robot for task j, n_j is the number of robots doing task j in strategy profile $\boldsymbol{s}$, $W_j^\star$ is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}^\star$,$\widetilde{W_j^\star}$ is the integer generated by applying the ceiling operator to $W_j^\star$, and $max\{n_j - 1, 1\}$ represents the larger number of $n_j - 1$ and 1. W_j and $W_j^\star$ are both ranging from 1 to n instead of ranging from 0 to n due to assumption 7 and 8.

Then we have:

$$\begin{aligned} J(\boldsymbol{s}) &\geq \sum_{\{j\in T|n_j^\star>0\}} l_j(\underset{\sim}{W_j})n_j^\star \\ &\geq \sum_{\{j\in T|n_j^\star>0\}} \lambda l_j(\widetilde{W_j^\star})n_j^\star - \mu l_j(\underset{\sim}{W_j})n_j \\ &\geq \sum_{\{j\in T|n_j^\star>0\}} \lambda l_j(W_j^\star)n_j^\star - \mu l_j(W_j)n_j \\ &\geq \sum_{\{j\in T\}} \lambda l_j(W_j^\star)n_j^\star - \mu l_j(W_j)n_j \\ &= \lambda J(\boldsymbol{s}^\star) - \mu J(\boldsymbol{s}) \end{aligned} \tag{4.15}$$

where $J(\boldsymbol{s}) = \sum_{i\in N} P_i(\boldsymbol{s})$, $J(\boldsymbol{s}^\star) = \sum_{i\in N} P_i(\boldsymbol{s}^\star)$, W_j is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}$, $W_j^\star$ is the sum of weights of task j's participating robots in strategy profile $\boldsymbol{s}^\star$, $n_j^\star$ is the number of robots doing task j in strategy profile $\boldsymbol{s}^\star$, n_j is the number of robots doing task j in strategy profile $\boldsymbol{s}$, $\underset{\sim}{W_j}$ is the integer generated by applying the floor operator to W_j and $\widetilde{W_j^\star}$ is the integer generated by applying the ceiling operator to $W_j^\star$. The first inequality holds due to (4.13). The second inequality is true due to (4.14). The third inequality holds as a result of the fact that for $\forall j \in T$, l_j is a monotonically increasing function for this specific resource allocation problem. The forth inequality holds as $\mu > 0$.

Hence, we get a lower bound of suboptimality for any given pure Nash equilibrium $\boldsymbol{s}$:

$$\alpha = \frac{J(\boldsymbol{s})}{J(\boldsymbol{s}^\star)} \geq \frac{\lambda}{1+\mu} \tag{4.16}$$

In the derivation of inequality (4.12), we assume after robot i move from s_i to $s_i^\star$, the resultant strategy profile still satisfies the constraint. However, it is possible to the resultant strategy profile does not satisfies the constraints. To handle this case, we let λ and μ satisfy additional constraints for arbitrary task $j \in T$:

$$\begin{aligned} l_j(\underset{\sim}{W_j}) \geq \lambda l_j(\widetilde{W_j^\star})n_j^\star - \mu l_j(\underset{\sim}{W_j})n_j \quad n_j = \{1, \ldots, c_j\}, \\ n_j^\star = \{1, \ldots, c_j\}, \quad \underset{\sim}{W_j} = \{1, ..., max\{n_j - 1, 1\}\}, \quad \widetilde{W_j^\star} = \{1, ..., n_j^\star\} \end{aligned} \tag{4.17}$$

Suppose there is a set $N^{'}$ of robots switching to task j in the strategy profile $s^{'}$, but the resultant strategy profile would violate the constraint. The reason why (4.16) works is as follows:

$$\begin{aligned} \sum_{i \in N'} P_i(\boldsymbol{s}) &= \sum_{i \in N'} l_{\boldsymbol{s}_i}(W_{\boldsymbol{s}_i}) \\ &\geq \lambda l_j(\widetilde{W_j^\star})n_j^\star - \mu l_j(\underset{\sim}{W_j})n_j \end{aligned} \tag{4.18}$$

where $\boldsymbol{s}_i$ is the task robot i chooses in the strategy profile $\boldsymbol{s}$. The inequality holds due to (4.17). We can plug (4.18) into (4.15) such that the suboptimality lower bound still holds true.

In summary, for this specific resource allocation problem, one way to find the lower bound of suboptimality is to solve the following nonlinear optimization problem with linear constraints:

$$\begin{aligned} \max_{\lambda>0,\mu>0} \quad & \frac{\lambda}{1+\mu} \\ s.t. \quad & \lambda W_q q - \mu W_p p \leq W_p q, \\ & p = \{1, \ldots, (c_k)_{max} - 1\}, q = \{1, \ldots, (c_k)_{max}\}, W_p = \{1, ..., max\{p^{'} - 1, 1\}\}, W_q = \{1, ..., q\} \\ & \lambda W_q^{'} q^{'} - \mu W_p^{'} p^{'} \leq W_p^{'}, \\ & p^{'} = \{1, \ldots, (c_k)_{max}\}, q^{'} = \{1, \ldots, (c_k)_{max}\}, W_p = \{1, ..., max\{p^{'} - 1, 1\}\}, W_q = \{1, ..., q\} \end{aligned}$$

where $(c_k)_{max}$ is the maximum of $c_k, \forall k \in T$ and c_j is the constraint on the maximum

number of participating robots of task j.

Example 3. *Suppose we have 5 tasks, and 18 robots belong to four different species. Each task requires at most four robots to participate. The suboptimality lower bound obtained via solving the nonlinear optimization problem is 0.0625.*

4.5 Strong Nash Equilibrium

In this section, We show that the set of strong Nash equilibria is not equivalent to the set of pure Nash equilibria for weighted congestion games in the context of ST-MR task allocation problem. By contrast, theorem 4 states that the set of strong Nash equilibria is equivalent to the set of pure Nash equilibria for standard congestion games in the context of ST-MR task allocation problem. We further discuss strong Nash equilibrium's potential to improve suboptimality lower bounds.

Theorem 6. *In the context of ST-MR task allocation problem, strong Nash equilibrium exists for weighted congestion games with monotonically decreasing latency functions [47].*

Strong Nash equilibria are pure Nash equilibria, but pure Nash equilibria are not necessarily strong Nash equilibrium. Next, we give an example to show a pure Nash equilibrium is not a strong Nash equilibrium for weighted congestion games in the context of ST-MR task allocation problem.

Table 4.1: weight profile for example 4

	Task 1	Task 2	Task 3	Task 4
species 1	0.68	0.5539	0.6440	0.6328
species 2	0.13	0.8908	0.4075	0.7593
species 3	0.0702	0.5858	0.2614	0.2339
species 4	0.0454	0.5315	0.6092	0.2887

Example 4. *Suppose we have four tasks, four species of robots and each species has ten*

Table 4.2: Example's task allocation result

	Task 1	Task 2	Task 3	Task 4
species 1	4	1	3	2
species 2	5	2	2	1
species 3	1	1	3	5
species 4	7	1	0	2

robots. Table 4.1 shows the weight profiles for each species of robots. Table 4.2 shows the task allocation result, which is a pure Nash equilibrium but not a strong Nash equilibrium.

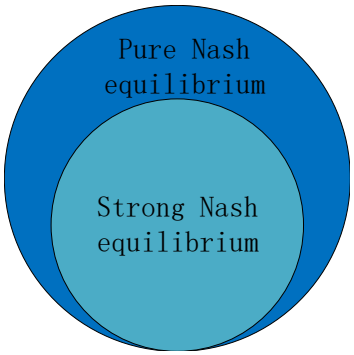


Figure 4.4: Strong Nash equilibrium is a subset of pure Nash equilibrium for weighted congestion games in the context of ST-MR task allocation problems

The motivation of investigating strong Nash equilibrium is straightforward. As shown in figure 4.4, strong Nash equilibrium is a subset of pure Nash equilbirum. Hence, strong Nash equilibria might be able to provide approximate solutions for resource allocation problems with higher suboptimality lower bounds compared to pure Nash equilibrium.

4.6 simulations

4.6.1 Scalability

We conduct numerical simulations to investigate sequential best response dynamics's scalability regarding the number of tasks, robots and species. We define an iteration as an loop of the main loop of the algorithms (lines 10-24 for sequential best response dynamics). For

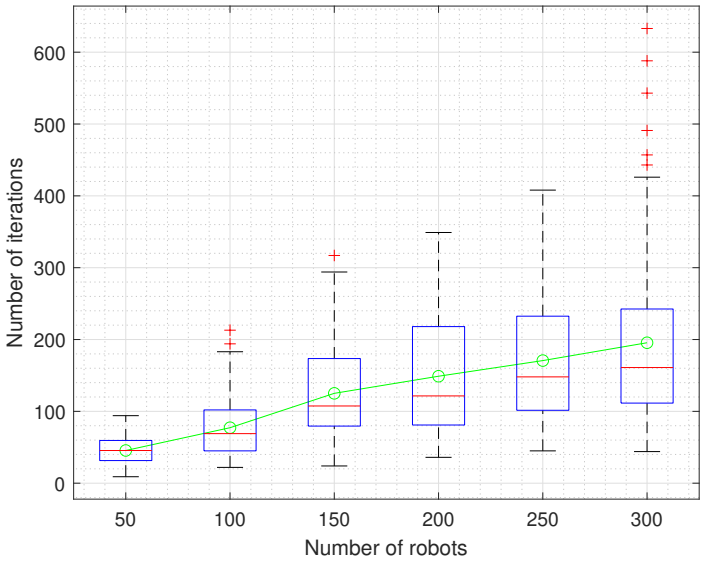


Figure 4.5: Convergence performance for sequential best response dynamics with decreasing marginal gain. Number of iterations of convergence for varying numbers of robots

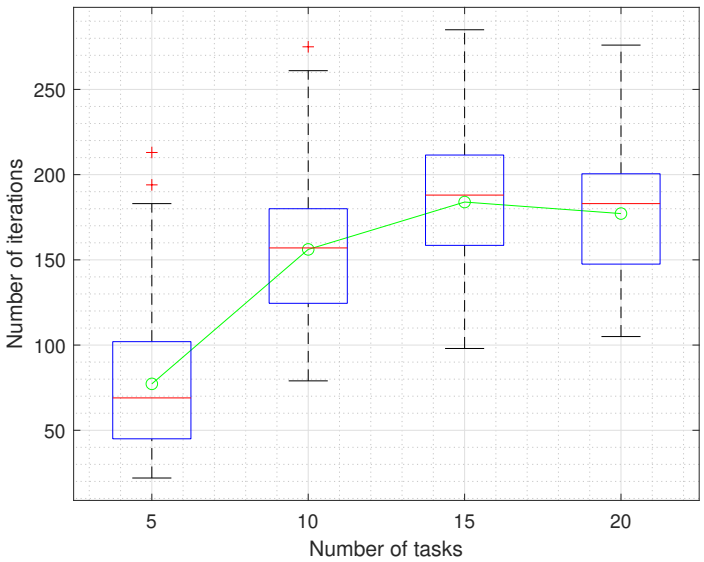


Figure 4.6: Convergence performance for sequential best response dynamics with decreasing marginal gain. Number of iterations of convergence for varying numbers of tasks

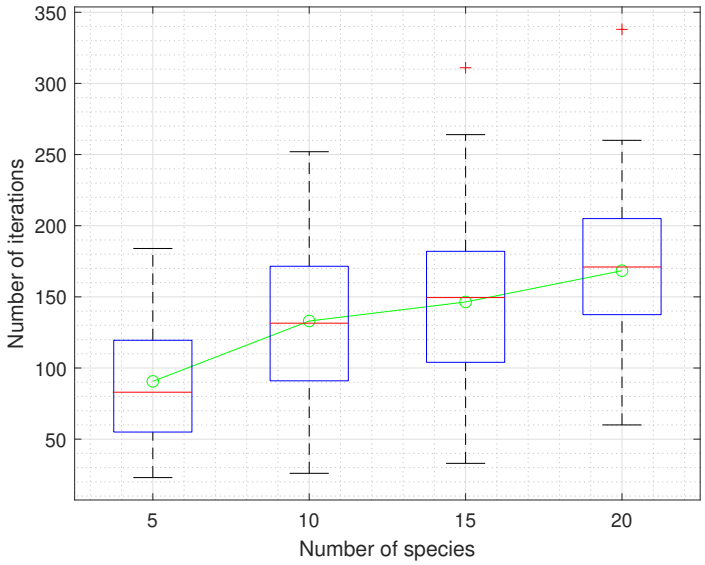


Figure 4.7: Convergence performance for sequential best response dynamics with decreasing marginal gain. Number of iterations of convergence for varying numbers of species of robots

each test case, we run 100 Monte Carlo trials. The statistical results of iterations required for convergences are shown in figures 4.5-4.10 as box plots. The mean values of iterations are indicated by a green circle and connected via green lines.

Figures 4.5-4.7 show convergence performances for sequential best response dynamics applying to resource allocation problem with decreasing marginal gain. Figure 4.5 presents the statistical results for varying numbers of robots. We fix the number of tasks and species to 5. The number of iterations for convergence is approximately a linear function of the number of robots. Figure 4.6 shows the statistical results for varying numbers of tasks. We fix the number of robots to 100 and species to 5. The number of iterations for convergence peaks at t=15 and then goes down at t=20. This is possibly due to the combinatorial time complexity in section 4.3.3. Figure 4.7 presents the statistical results for varying numbers of species. We fix the number of robots to 120 and the number of tasks to 5. Each species of robots have the same number of robots. The number of iterations for convergence is approximately a linear function of the number of robots.

Figures 4.8-4.10 show convergence performances for sequential best response dynam-

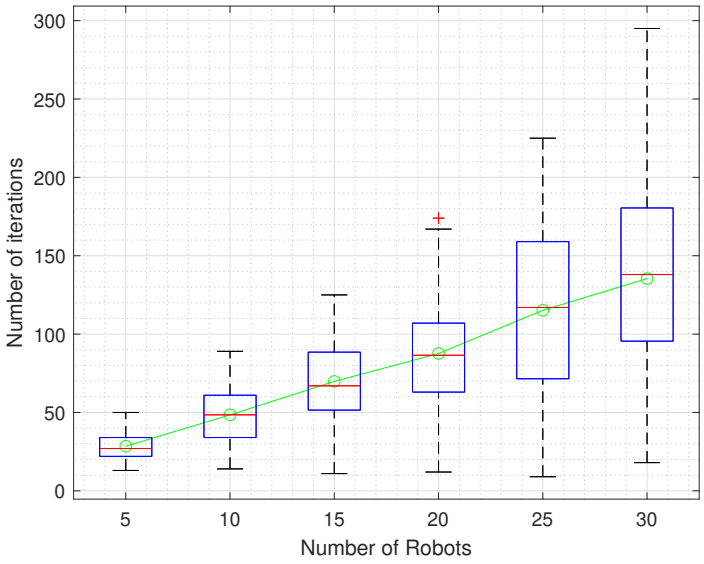


Figure 4.8: Convergence performance for sequential best response dynamics with increasing marginal gain. Number of iterations of convergence for varying numbers of robots

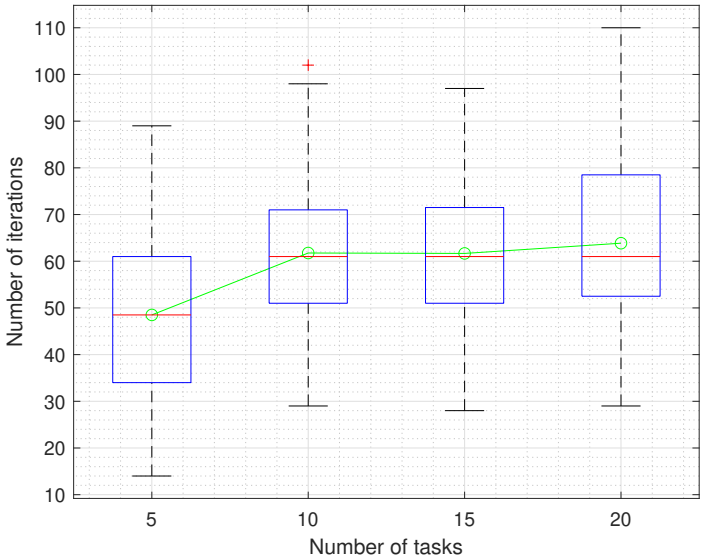


Figure 4.9: Convergence performance for sequential best response dynamics with increasing marginal gain. Number of iterations of convergence for varying numbers of tasks

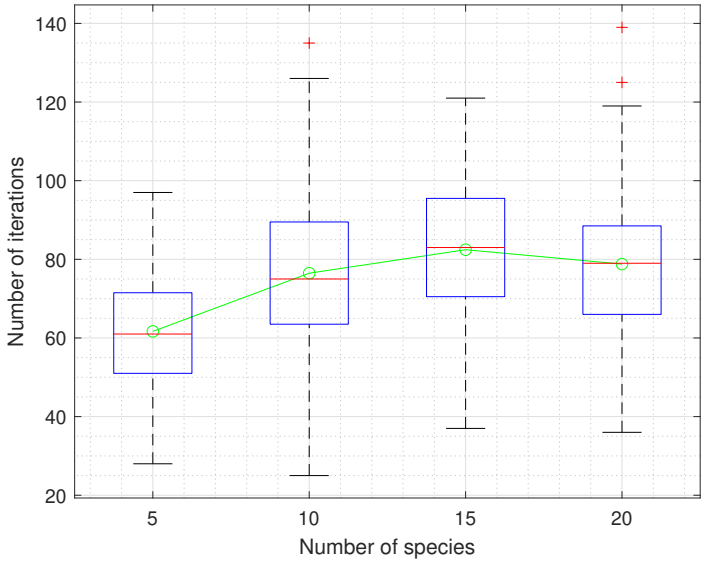


Figure 4.10: Convergence performance for sequential best response dynamics with increasing marginal gain. Number of iterations of convergence for varying numbers of species of robots

ics applying to resource allocation problem with increasing marginal gain. Figure 4.8 presents the statistical results for varying numbers of robots. We fix the number of tasks and species to 5. The number of iterations for convergence is approximately a linear function of the number of robots. Figure 4.9 shows the statistical results for varying numbers of tasks. We fix the number of robots to 100 and species to 5. The number of iterations bottoms at t=5 while remaining constant for t=5,15,20. Figure 4.7 presents the statistical results for varying numbers of species. We fix the number of robots to 100 and the number of tasks to 5. The number of iterations for convergence is approximately a linear function of the number of robots. This is possibly due to the combinatorial time complexity in section 4.3.3.

4.6.2 Suboptimality

We conduct 1000 Monte Carlo trials to investigate suboptimality performances for both types of resource allocation problems. Tests results are presented in figures 4.11 and 4.12 as histograms.

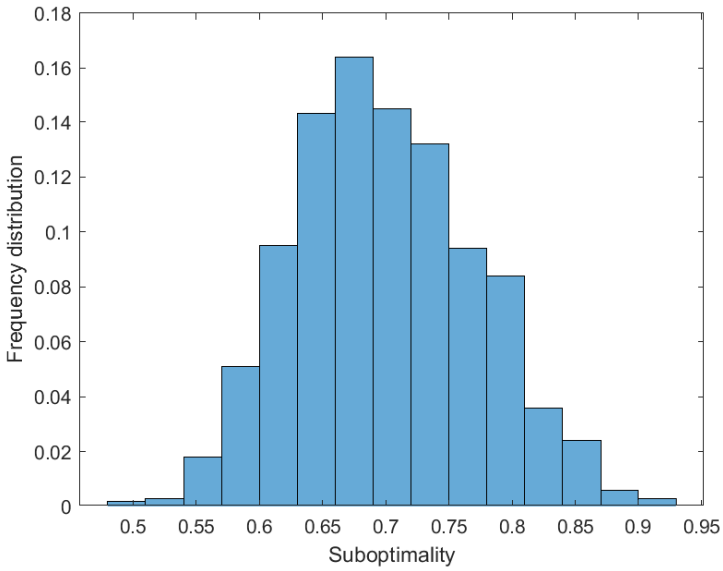


Figure 4.11: Suboptmality performances for resource allocation problems with decreasing marginal gains

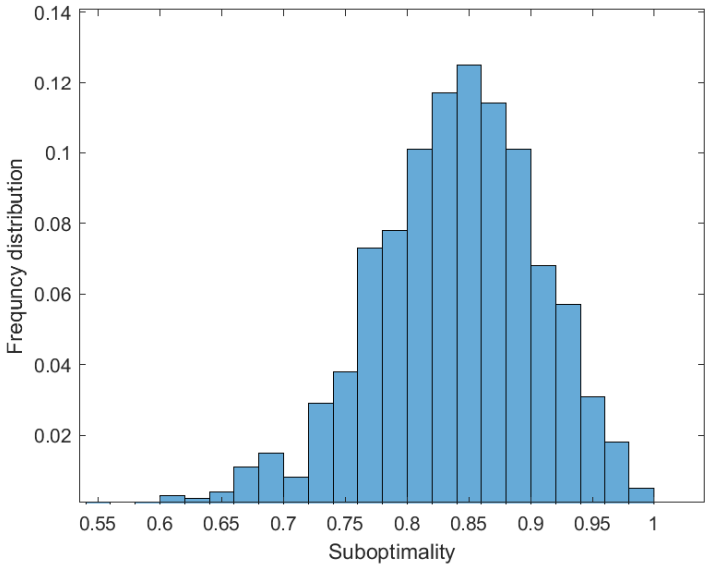


Figure 4.12: Suboptmality performances for resource allocation problems with increasing marginal gains

For resource allocation problems with decreasing marginal gains, we consider a scenario of 20 robots belong to four species and four different tasks. Each species has five robots. As shown in figure 4.11, the frequency distribution is peaked for suboptimality ranging from 06.-0.7. The minimum suboptimality obtained in the experiment is 0.49.

For resource allocation problems with increasing marginal gains, we consider a scenario of 18 robots belong to three species and four different tasks. Each species has six robots. As shown in figure 4.12, the frequency distribution is peaked for suboptimality ranging from 08.-0.9. The minimum suboptimality obtained in the experiment is 0.59, while the suboptimality lower bound obtained by following the approach in section 4.4.1 is 0.0625.

4.7 Experiments on Robotarium

In this section, we implement decentralized sequential best response dynamics for both types of resource allocation problems on Robotarium.

For resource allocation problems with decreasing marginal gains, we consider a scenario with seven robots belong to 4 species and four tasks. We divide the Robotarium into 4 subareas. Each subarea represents a task. If a robot is staying in a subarea, it is participating in the task which this specific subarea represents. Species 1,2, and 3 each have two robots. Species 4 has a single robot. Weights (priorities) for each task are 0.8, 0.7, 0.75,0.9. Weight profiles for each species of robots are shown in table 4.3. Task allocation results are shown in table 4.4 and figure 4.13.

Table 4.3: weight profiles for decreasing marginal gain

	Task 1	Task 2	Task 3	Task 4
species 1	0.3962	0.2417	0.9412	0.2348
species 2	0.1112	0.4039	0.9561	0.3532
species 3	0.7803	0.0965	0.5752	0.8212
species 4	0.3897	0.1320	0.0598	0.0154

For resource allocation problems with increasing marginal gains, we consider a scenario

Table 4.4: task allocation result for decreasing marginal gain

	Task 1	Task 2	Task 3	Task 4
species 1	1	0	0	1
species 2	1	0	0	1
species 3	0	2	0	0
species 4	0	0	1	0

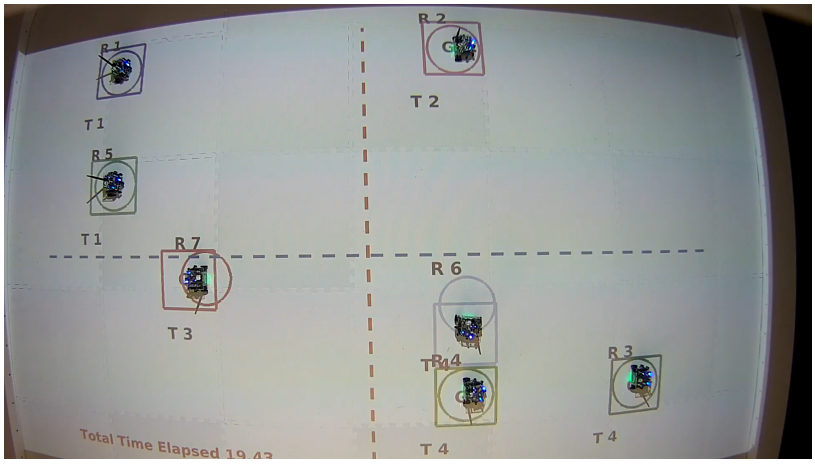


Figure 4.13: task allocation result for decreasing marginal gain

with seven robots belong to 4 species and four tasks. Species 1,2, and 3 each have two robots. Species 4 has a single robot. Weights (priorities) for each task are 0.8, 0.7, 0.75,0.9. The maximum numbers of participating robots for each task are 2,3,2,3. Weight profiles for each species of robots are shown in table 4.5. Task allocation results are shown in table 4.6 and figure 4.14.

Table 4.5: weight profiles for increasing marginal gain

	Task 1	Task 2	Task 3	Task 4
species 1	0.3909	0.9976	0.1375	0.7316
species 2	0.0546	0.8116	0.3900	0.6183
species 3	0.5013	0.4857	0.9274	0.3433
species 4	0.4317	0.8944	0.9175	0.9360

Table 4.6: task allocation result for increasing marginal gain

	Task 1	Task 2	Task 3	Task 4
species 1	1	1	0	0
species 2	0	1	0	1
species 3	0	0	2	0
species 4	0	1	0	0

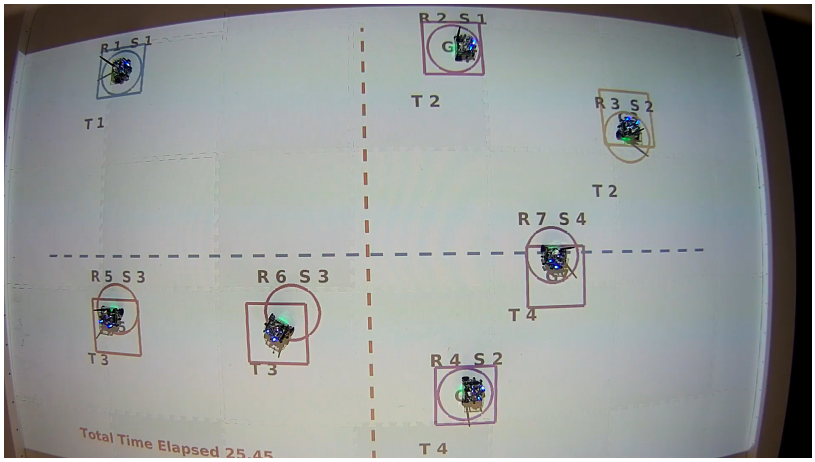


Figure 4.14: task allocation result for increasing marginal gain

CHAPTER 5
CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this thesis, we aim to develop decentralized task allocation algorithms to provide approximate solutions to resource allocation problems. We consider two types of resource allocation problems, with monotonically increasing marginal gains or with monotonically decreasing marginal gains. Thanks to the hierarchical structure of congestion games, we can decompose the global level objective functions to robot level payoff functions. For homogeneous robot teams, we unify the decentralized best response dynamics under the framework of standard congestion game theory. We also propose decentralized concurrent best response dynamics and decentralized dual greedy algorithm. For heterogeneous robot teams, we show decentralized best response dynamics would converge to a pure Nash equilibrium. For both homogeneous and heterogeneous robot teams, we discuss the suboptimality of pure Nash equilibria via $\lambda-\mu$ smoothness technique. Simulations are conducted to investigate scalability and suboptimality. The above algorithms are implemented on the Robotraium.

5.2 Future Work

We point out three interesting directions for future work.

Suboptimality lower bound In the experiments, we observe that suboptimality lower bounds obtained by $\lambda - \mu$ smoothness argument is conservative. We observe much better performances in experiments compared to the obtained suboptimality lower bounds. Hence, it would be useful to obtain tighter suboptimality lower bounds.

Strong Nash equilibrium In section 4.5, we discuss strong Nash equilibrium's poten-

tial to improve suboptimality lower bounds. It would be interesting to generalize the dual greed algorithm to weighted congestion games or develop other algorithms to converge to strong Nash equilibrium.

Interference games It is possible that different species of robots could cooperate in complementing each other to achieve better performances compared to working alone. To capture this characteristic, we could consider adding an interference structure to congestion games. To be more specific, the total weight for a task is not a simple sum of all participating robot's weights, but a function of the composition of robot teams.

REFERENCES

[1] G.-Z. Yang, J. Bellingham, P. E. Dupont, P. Fischer, L. Floridi, R. Full, N. Jacobstein, V. Kumar, M. McNutt, R. Merrifield, *et al.*, "The grand challenges of science robotics," *Science robotics*, vol. 3, no. 14, 2018.

[2] G.-Z. Yang, B. J. Nelson, R. R. Murphy, H. Choset, H. Christensen, S. H. Collins, P. Dario, K. Goldberg, K. Ikuta, N. Jacobstein, *et al.*, *Combating covid-19—the role of robotics in managing public health and infectious diseases*, 2020.

[3] S. Manjanna, A. Q. Li, R. N. Smith, I. Rekleitis, and G. Dudek, "Heterogeneous multi-robot system for exploration and strategic water sampling," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 1–8.

[4] N. Michael, S. Shen, K. Mohta, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, K. Ohno, *et al.*, "Collaborative mapping of an earthquake damaged building via ground and aerial robots," in *Field and service robotics*, Springer, 2014, pp. 33–47.

[5] L. Luo, "Distributed algorithm design for constrained multi-robot task assignment," PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 2014.

[6] R. W. Rosenthal, "A class of games possessing pure-strategy nash equilibria," *International Journal of Game Theory*, vol. 2, no. 1, pp. 65–67, 1973.

[7] S. Wilson, P. Glotfelter, L. Wang, S. Mayya, G. Notomista, M. Mote, and M. Egerstedt, "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Systems Magazine*, vol. 40, no. 1, pp. 26–44, 2020.

[8] I. Jang, H.-S. Shin, and A. Tsourdos, "Anonymous hedonic game for task allocation in a large-scale multiple agent system," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1534–1548, 2018.

[9] H. Ackermann, P. Berenbrink, S. Fischer, and M. Hoefer, "Concurrent imitation dynamics in congestion games," in *Proceedings of the 28th ACM symposium on Principles of distributed computing*, 2009, pp. 63–72.

[10] T. Roughgarden, "Intrinsic robustness of the price of anarchy," *Journal of the ACM (JACM)*, vol. 62, no. 5, pp. 1–42, 2015.

[11] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International journal of robotics research*, vol. 23, no. 9, pp. 939–954, 2004.

[12] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1495–1512, 2013.

[13] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[14] R. E. Marsten and F. Shepardson, "Exact solution of crew scheduling problems using the set partitioning model: Recent successful applications," *Networks*, vol. 11, no. 2, pp. 165–177, 1981.

[15] A. Atamtürk, G. L. Nemhauser, and M. W. Savelsbergh, "A combined lagrangian, linear programming, and implication heuristic for large-scale set partitioning problems," *Journal of heuristics*, vol. 1, no. 2, pp. 247–259, 1996.

[16] M. S. Rasmussen, "Optimisation-based solution methods for set partitioning models," 2011.

[17] D. Y. Yeh, "A dynamic programming approach to the complete set partitioning problem," *BIT Numerical Mathematics*, vol. 26, no. 4, pp. 467–474, 1986.

[18] R. S. Garfinkel and G. L. Nemhauser, "The set-partitioning problem: Set covering with equality constraints," *Operations Research*, vol. 17, no. 5, pp. 848–856, 1969.

[19] R. E. Marsten, "An algorithm for large set partitioning problems," *Management Science*, vol. 20, no. 5, pp. 774–787, 1974.

[20] A. Caprara, P. Toth, and M. Fischetti, "Algorithms for the set covering problem," *Annals of Operations Research*, vol. 98, no. 1-4, pp. 353–371, 2000.

[21] M. B. Dias, "Traderbots: A new paradigm for robust and efficient multirobot coordination in dynamic environments," *Robotics Institute*, p. 153, 2004.

[22] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for multirobot coordination," *IEEE transactions on robotics and automation*, vol. 18, no. 5, pp. 758–768, 2002.

[23] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE transactions on robotics and automation*, vol. 14, no. 2, pp. 220–240, 1998.

[24] Y. Hatano and M. Mesbahi, "Agreement over random networks," *IEEE Transactions on Automatic Control*, vol. 50, no. 11, pp. 1867–1872, 2005.

[25] C. W. Wu, "Synchronization and convergence of linear dynamics in random directed networks," *IEEE transactions on Automatic control*, vol. 51, no. 7, pp. 1207–1210, 2006.

[26] A. Tahbaz-Salehi and A. Jadbabaie, "On consensus over random networks," in *44th Annual Allerton Conference*, Citeseer, 2006.

[27] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE transactions on robotics*, vol. 25, no. 4, pp. 912–926, 2009.

[28] L. Luo, N. Chakraborty, and K. Sycara, "Provably-good distributed algorithm for constrained multi-robot task assignment for grouped tasks," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 19–30, 2014.

[29] L. Lin and Z. Zheng, "Combinatorial bids based multi-robot task allocation method," in *Proceedings of the 2005 IEEE international conference on robotics and automation*, IEEE, 2005, pp. 1145–1150.

[30] L. Vig and J. A. Adams, "Multi-robot coalition formation," *IEEE transactions on robotics*, vol. 22, no. 4, pp. 637–649, 2006.

[31] O. Shehory and S. Kraus, "Methods for task allocation via agent coalition formation," *Artificial intelligence*, vol. 101, no. 1-2, pp. 165–200, 1998.

[32] Z. Han, D. Niyato, W. Saad, T. Başar, and A. Hjørungnes, *Game theory in wireless and communication networks: theory, models, and applications*. Cambridge university press, 2012.

[33] M. Zulhasnine, C. Huang, and A. Srinivasan, "Efficient resource allocation for device-to-device communication underlaying lte network," in *2010 IEEE 6th International conference on wireless and mobile computing, networking and communications*, IEEE, 2010, pp. 368–375.

[34] R. Buyya, D. Abramson, J. Giddy, *et al.*, "An economy driven resource management architecture for global computational power grids.," in *PDPTA*, 2000, pp. 26–29.

[35] R. Wolski, J. S. Plank, T. Bryan, and J. Brevik, "G-commerce: Market formulations controlling resource allocation on the computational grid," in *Proceedings 15th International Parallel and Distributed Processing Symposium. IPDPS 2001*, IEEE, 2000, 8–pp.

[36] Y. Ge, Y. Zhang, Q. Qiu, and Y.-H. Lu, "A game theoretic resource allocation for overall energy minimization in mobile cloud computing system," in *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*, 2012, pp. 279–284.

[37] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE transactions on parallel and distributed systems*, vol. 24, no. 6, pp. 1107–1117, 2012.

[38] K. M. Bretthauer and B. Shetty, "The nonlinear knapsack problem–algorithms and applications," *European Journal of Operational Research*, vol. 138, no. 3, pp. 459–472, 2002.

[39] H. Bayram and H. I. Bozma, "Coalition formation games for dynamic multirobot tasks," *The International Journal of Robotics Research*, vol. 35, no. 5, pp. 514–527, 2016.

[40] H. Ackermann, "Nash equilibria and improvement dynamics in congestion games," PhD thesis, Ph. D. thesis, RWTH Aachen University, 2009.

[41] D. Monderer and L. S. Shapley, "Potential games," *Games and economic behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[42] P. N. Panagopoulou and P. G. Spirakis, "Algorithms for pure nash equilibria in weighted congestion games," *Journal of Experimental Algorithmics (JEA)*, vol. 11, pp. 2–7, 2007.

[43] E. Even-Dar, A. Kesselman, and Y. Mansour, "Convergence time to nash equilibria," in *International Colloquium on Automata, Languages, and Programming*, Springer, 2003, pp. 502–513.

[44] G. Christodoulou and E. Koutsoupias, "The price of anarchy of finite congestion games," in *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, 2005, pp. 67–73.

[45] T. Roughgarden, "Intrinsic robustness of the price of anarchy," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 513–522.

[46] S. Aland, D. Dumrauf, M. Gairing, B. Monien, and F. Schoppmann, "Exact price of anarchy for polynomial congestion games," *SIAM Journal on Computing*, vol. 40, no. 5, pp. 1211–1233, 2011.

[47] T. Harks, M. Klimm, and R. H. Möhring, "Strong equilibria in games with the lexicographical improvement property," *International Journal of Game Theory*, vol. 42, no. 2, pp. 461–482, 2013.

[48] S. Chien and A. Sinclair, "Strong and pareto price of anarchy in congestion games," in *International Colloquium on Automata, Languages, and Programming*, Springer, 2009, pp. 279–291.

[49] N. Andelman, M. Feldman, and Y. Mansour, "Strong price of anarchy," *Games and Economic Behavior*, vol. 65, no. 2, pp. 289–317, 2009.

[50] M. Ibrahim, K. Khawam, and S. Tohme, "Congestion games for distributed radio access selection in broadband networks," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, IEEE, 2010, pp. 1–5.

[51] E. Biyik, D. A. Lazar, R. Pedarsani, and D. Sadigh, "Altruistic autonomy: Beating congestion on shared roads," in *Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2018.

[52] L. Zhou and P. Tokekar, "Sensor assignment algorithms to improve observability while tracking targets," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1206–1219, 2019.

[53] M. Corah and N. Michael, "Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice," *Autonomous Robots*, vol. 43, no. 2, pp. 485–501, 2019.

[54] S. Ieong, R. McGrew, E. Nudelman, Y. Shoham, and Q. Sun, "Fast and compact: A simple class of congestion games," in *AAAI*, vol. 5, 2005, pp. 489–494.

[55] T. Harks, M. Hoefer, M. Klimm, and A. Skopalik, "Computing pure nash and strong equilibria in bottleneck congestion games," *Mathematical Programming*, vol. 141, no. 1-2, pp. 193–215, 2013.

[56] R. Holzman and N. Law-Yone, "Strong equilibrium in congestion games," *Games and economic behavior*, vol. 21, no. 1-2, pp. 85–101, 1997.